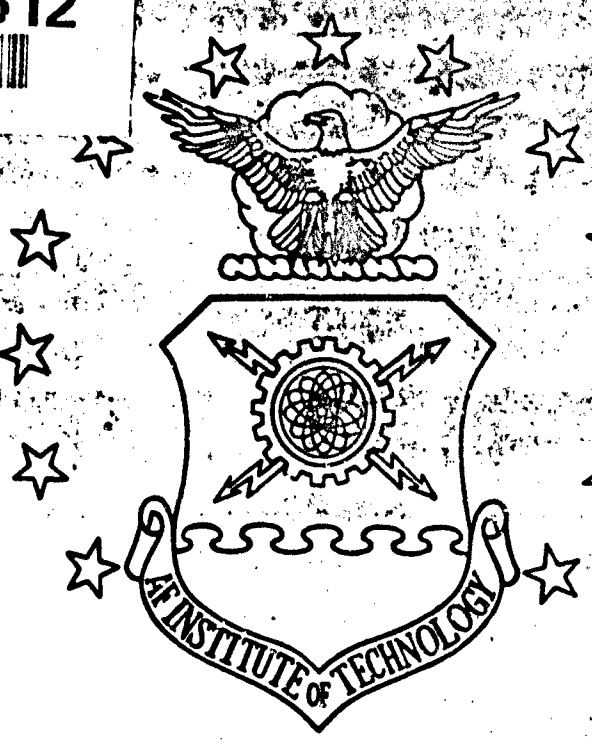


A COMPARISON OF VARIABLE SELECTION CRITERIA FOR MULTIPLE LINEAR REGRESSION: A SECOND SIMULATION STUDY

THESIS

David P. Woollard, Captain, USAF

AFIT/GOR/ENC/93M-22

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY
AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

AFIT/GOR/ENS/93M-23

A COMPARISON OF VARIABLE SELECTION CRITERIA FOR MULTIPLE LINEAR REGRESSION: A SECOND SIMULATION STUDY

THESIS

David P. Woollard, Captain, USAF

AFIT/GOR/ENS/93M-23

Approved for public release; distribution unlimited

AFIT/GOR/ENS/93M-23

A COMPARISON OF VARIABLE SELECTION CRITERIA FOR MULTIPLE LINEAR REGRESSION: A SECOND SIMULATION STUDY

THESIS

Presented to the Faculty of the School of Engineering
of the Air Force Institute of Technology
Air University
In Partial Fulfillment of the
Requirements for the Degree of
Master of Science in Operations Research

David P. Woollard, B.S., M.L.A.
Captain USAF

March, 1993

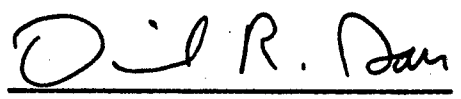
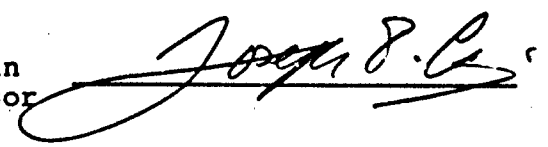
Approved for public release; distribution unlimited

THESIS APPROVAL

STUDENT: Captain David P. Woollard CLASS: GOR 93-M

THESIS TITLE: A Comparison of Variable Selection Criteria for Multiple Linear Regression: A Second Simulation Study

DEFENSE DATE: 10 MAR 93

COMMITTEE	NAME/DEPARTMENT	SIGNATURE
Advisor	Dr. David R. Barr Associate Professor ENC	
ENS Rep.	Dr. Joseph P. Cain Associate Professor ENS	

Acknowledgements

I would like to express my deepest thanks to Dr. Barr who always kept me on the right course. He told me that when one starts a research project, one never can be certain of where he or she may end up. This thesis, then, is a map of where I went and how I got there.

I would not have completed this project without the assistance of my academic partner and wife, Karen. Her encouragement, understanding, and suggestions were much appreciated.

Most of all, I'd like to thank Jesus Christ for creating the beautiful laws of mathematics. He held the universe and my sanity together long enough for me to accomplish this project (Colossians 1:16-17). I'll be grateful to Him throughout eternity.

David P. Woollard

Table of Contents

List of Figures

List of Tables

Abstract

The purpose of this thesis was to: (1) identify some promising least squares selection procedures discussed in the literature, (2) introduce, implement, and study a variable selection method proposed by Alan J. Miller, and, (3) make an extension of Ross J. Hansen's 1988 thesis research by comparing the methods he examined: Minimum MSE, Minimum S_p, and Minimum C_p with Miller's method.

To expedite a comparative analysis of Miller's method and the other methods, Response Surface methodology was employed with two performance measures. The first was the percentage of correct variables in a model. The second, the Theoretical Mean Squared Error of Prediction (TMSEP), measured the predictive error between the model selected and the theoretical model. Each technique was applied on generated data with known multicollinearities, variances, random predictors, and sample sizes. Both performance measures were computed for models selected under each technique. A full factorial design using each performance measure was set up to study the effectiveness of each variable selection technique with respect to the known data characteristics. Equations were generated which related these data characteristics to each combination of perfor-

mance measure and selection method. A graphical analysis of variance was performed to summarize each technique's performance.

Miller's method was shown to be the best overall technique for selecting models with the highest percentage of correct variables. Minimum MSE, followed closely by Minimum S_p, selected models with the least TMSEP.

A COMPARISON OF VARIABLE SELECTION CRITERIA FOR MULTIPLE LINEAR REGRESSION: A SECOND SIMULATION STUDY

I. Introduction

Background

Linear regression is a statistical model-building tool that uses data to construct a mathematical expression capable of estimating the actual, but unknown, relationship between a set of independent or predictor variables and their corresponding response values. This mathematical expression or model can, with a certain degree of accuracy, predict the level of response of the associated phenomena, given a set of predictor values. The methodologies, processes and techniques employed to select which predictor variables to include in a model form a sub-topic of linear regression called subset selection. Unfortunately, it is often difficult to determine the "best" set of predictor variables to include in a linear regression model (Hansen, 1988:1). Alan J. Miller, an expert in the field of subset selection, assesses the situation on the back cover of his newest book, Subset Selection in Regression:

> Most scientific computing packages contain facilities for stepwise regression, and often for "all subsets" and other techniques for finding "best-fitting" subsets of regression variables.

> The application of standard theory can be very misleading in such cases when the model has not been chosen a priori, but from the data. There is widespread awareness that considerable overfitting occurs, and that prediction equations obtained after extensive "data dredging" often perform poorly when applied to new data. (Miller, 1990:cover)

Clearly, as A.J. Miller points out, automated subset selection processes are not foolproof. Over-fitting is likely when one blindly applies an automated subset selection method, such as Stepwise Regression, to data containing both significant and insignificant (random) predictors. The automated software selects predictors on the basis of some preset criteria and will probably find the "best fit" when a large number of these predictors are included in the model, including any random ones. At first, it may seem that predictors that are theoretically independent of the response would not be selected because they contribute nothing to the response. Freedman, however, demonstrated that this is not necessarily the case. His research indicates a good fit could result even when a model is constructed from only random noise predictors (Freedman, 1983:153). Furthermore, when automatic algorithms compare models containing only significant predictors and those containing the same significant predictors augmented with random predictors, one of the models containing randomness is often selected. This occurs because the largest sample correlation among the random predictors can become significant and, in turn, cause

the automated algorithm to favor a model composed of both significant and random predictors.

The problem of over-fitting emphasizes that when the subset selection process is blindly turned over to automated algorithms implemented by computer software packages, the resulting mathematical equation may be useless. It may model the data and the noise in the data very well while failing to achieve the real goal of modeling the underlying process or phenomena. As a result, when one uses an over-fitted model to predict future response levels, it generally performs poorly because the presence of the random predictors effectively mask whatever predictive insight the significant predictors have to offer (Cafarella, 1979:14). Sometimes human judgement, tempered by years of experience, can recognize when over-fitting has occurred, can discontinue the automatic algorithm, and can select a more parsimonious model. Which criteria to use, however, in selecting a more parsimonious model that will indeed adequately represent the underlying process or phenomena may not be readily known. Obviously, research is needed to determine which subset selection criteria perform best under a given set of circumstances.

Problem Statement

This research effort collected and analyzed data on certain subset selection techniques to better understand why they perform as they do. The objectives of this research were: (1) identify some promising least squares selection procedures discussed in the literature, (2) introduce, implement, and study a variable selection method proposed by Alan J. Miller, and, (3) make an extension of Ross Hansen's 1988 thesis research by comparing the methods he examined: Minimum MSE, Minimum S_p, and Minimum C_p with Miller's method.

Assumptions

Miller's technique requires certain assumptions prior to its application. First, the collected data must be a random sample from the population. Next the error terms of the least squares linear regression must be independent and identically distributed, be from a normal distribution, and have a mean of zero and constant variance. Finally, when the Stepwise regression is run, only Forward Selection is used with a threshold F-value low enough to allow the selection of at least one known random predictor.

Scope

This study is an extension of Ross Hansen's research in which he examines three subset selection criteria (Minimum

MSE, Minimum Sp, and Minimum Cp) under varying amounts of multicollinearity, variable variation, number of variables, and sample size. Additionally, this study examined the performance of yet a fourth subset selection criteria, previously described and referred to as Miller's method, under the same conditions. The performance of Miller's method is compared to the performance of the three other criteria Hansen studied. Contrasts and comparisons are made and conclusions are drawn.

II. Concept Overview

Least Squares Regression

Assumptions. Certain key assumptions must be made prior to constructing a least squares linear regression. One must first assume the collected data represents the population from which it came. That is, the data reflects the normal case of the variable. Secondly, the error terms are assumed to be independent and identically distributed, from a normal distribution with a mean of zero and variance σ^2.

Notation. The aim of linear regression is to calculate what proportion of the independent variables should be added or subtracted to best predict the dependent variable. In general, the linear least squares regression equation is written:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_k X_k + e \qquad (1)$$

where:

Y is the observed value of the independent variable

β_0 is the constant term

$\beta_1, \beta_2, \ldots, \beta_k$ are constant proportional multipliers of the dependent variables $X_1, X_2, \ldots, X_k$

k is the number of independent variables

e is the error term.

If there are n observations, or data points, the above equation may be written as:

$$\sum_{i=1}^{n} Y_i = \sum_{i=1}^{n} (\beta_{0i} + \beta_{1i}X_{1i} + \beta_{2i}X_{2i} + \ldots + \beta_{ki}X_{ki} + e_i) \qquad (2)$$

For convenience, the above equation can be written in matrix notation.

$$\mathbf{Y} = \mathbf{X}\boldsymbol{\beta} + \mathbf{e} \qquad (3)$$

where Y is an nx1 column vector:

$$\mathbf{Y} = \begin{bmatrix} Y_1 \\ Y_2 \\ \cdot \\ \cdot \\ \cdot \\ Y_n \end{bmatrix} \qquad (4)$$

and X is a {nx(k+1)} matrix.

$$\mathbf{X} = \begin{bmatrix} 1 & X_{11} & X_{12} & \cdot & \cdot & \cdot & X_{1k} \\ 1 & X_{21} & X_{22} & \cdot & \cdot & \cdot & X_{2k} \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ 1 & X_{n1} & X_{n2} & \cdot & \cdot & \cdot & X_{nk} \end{bmatrix} \qquad (5)$$

The first column contains all ones for the constant terms. The remaining columns contain the X_{ij} independent variables. The **X** matrix is commonly referred to as the design matrix. **B** is a k×1 column vector of regression coefficient:

$$\boldsymbol{B} = \begin{bmatrix} B_0 \\ B_1 \\ \cdot \\ \cdot \\ \cdot \\ B_k \end{bmatrix} \tag{6}$$

and **e** is a n×1 column vector of error terms:

$$\boldsymbol{e} = \begin{bmatrix} e_1 \\ e_2 \\ \cdot \\ \cdot \\ \cdot \\ e_n \end{bmatrix} \tag{7}$$

In least squares regression, each subset of regression variables generates a surface which minimizes the squared distance (error) between the observed values for the dependent variables, Y, and the predicted values for the dependent variable,

$$\min \sum_{i=1}^{n} (e_i)^2 = \min \sum_{i=1}^{n} (\overline{Y}_i - Y_i)^2 \tag{8}$$

The goal is to find the subset of variables which minimizes the squared distances between the actual values observed and the fitted surface. The sum of the squared-error values is commonly referred to as the sum-of-squares error (SSE). Graphically, a regression resembles the following:

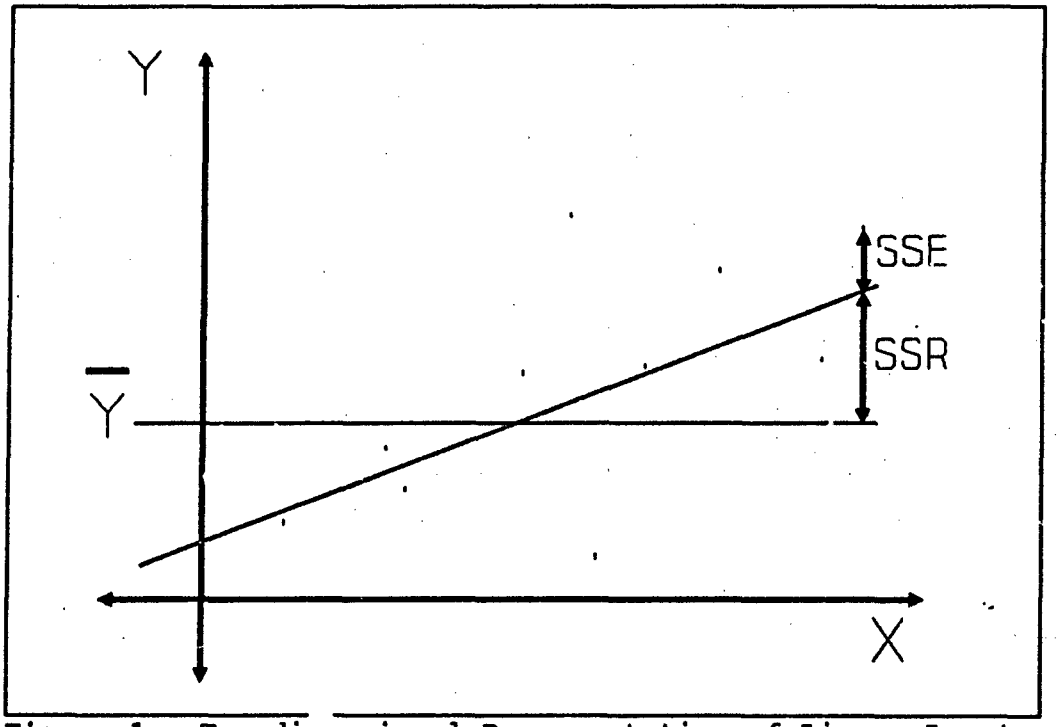


Figure 1. Two-dimensional Representation of Linear Least Squares Regression

SSR is the sum of the squared distances from the mean to the regression line, called Regression Sum of Squares. SSE is the sum of the squared distances from the point to the regression line, called Sum of Squares Error. SST is called Sum of Squares Total and is calculated by:

$$SST = SSR + SSE \qquad (9)$$

III. Summary of Current Knowledge

Regression

Regression Analysis, as a branch of statistical mathematics, began in the late 1800's when Sir Francis Galton first attempted to use practical mathematical techniques to investigate the dependence between two variables: the height of the parents (he used the average of the parents heights) and the heights of their adult children. Having randomly collected (sampled) many pairs of parent/child height measurements (data), Galton observed that for a given parent-height average, the conditional mean of the heights of children with that given average parent height "regressed" toward the mean height of all children. Thus, the term regression analysis was born (Neter and others, 1990:26). Regression techniques have since been developed that can construct an equation or mathematical model based on past historical data and then use this model to predict future responses (Neter and others, 1990:27).

Subset Selection

Subset selection is an area of regression analysis concerned with choosing the "best" variables, or predictors, to include in the regression model (Hocking, 1983:220). The simple parent/child height model yielded only two choices:

one could accept the model or reject it. Accepting the model meant that if one specified the parent-height, an estimate of the adult child-height was automatically generated. Consider, however, the complexity that occurs if one not only possesses the heights of the parents but also their right arm lengths. Then one would have to decide whether to use a model to predict adult child-height based on the height of the parents, or the right arm length of the parents, or both, or neither. The methodologies of subset selection can help suggest which predictors to use. Unfortunately, and as previously addressed, applying these methodologies, without discretion, has a tendency to produce over-fitted models that have little predictive capability (Miller, 1990:12-13).

In spite of these difficulties, however, subset selection does play an important role in regression analysis. While other areas of regression analysis detect and correct problems in the data prior to model creation or verify the adequacy of the model after creation, subset selection techniques actually select the variables or predictors that go into the model. These techniques are subdivided into two major groupings:

(1)Least Squares regression techniques

(2)Biased regression techniques

For this literature review, only the Least Squares regression techniques will be addressed. Selection techniques for

least squares have an advantage over bias regression techniques in that the estimators are the best linear unbiased estimators (BLUE).

All-subsets regression. All-subsets regression does just that--it forms a regression model for each predictor or combination of predictors. Miller claims that only by an exhaustive search of all 2^k-1 combinations or subsets can one be guaranteed to find the best-fitting model (Miller, 1984:391). Once generated, various criteria may be employed in searching all 2^k-1 models for the one that best fits the data. The all-subsets variable selection criteria addressed in this literature review are:

(1) Near-Optimal-Model for Mean Square Absolute Errors (MSAE),

(2) Mallows Cp,

(3) Coefficient of Determination or R^2,

(4) Maximum Adjusted R^2 or Minimum MSE,

(5) $PRESS_p$(Prediction Sum of Squares) or S_p.

Although exhaustive and guaranteed to find the "best" model ("best" being defined by the criteria used), the All-Subsets method has two drawbacks, regardless of the criteria involved. First, it can only be used for a moderately small number

of predictors because the number of possible subsets of predictors almost doubles with each additional predictor considered (e.g. 1 for 1 predictor, 3 for 2, 7 for 3, 524?87

for 19, 1048576 for 20, 33.5 million for 25, etc.)(Miller, 1990:56). Consequently, when considering a realistic number of predictors (15 to 25) one is forced to use a less exhaustive, but more efficient, subset selection technique such as Stepwise regression. Secondly, All-Subsets regression is only guaranteed to find the "best" model if all significant predictors are considered (Narula, 1983:160). If the group of predictors under consideration does not contain all the significant predictors, then the All-Subsets approach can not find the "best" overall model, but will produce the "best" model for the predictors considered (Berk, 1978:3).

Mallows C_p. Mallows C_p is a statistic used to determine the best model when the independent variables are fixed. C_p is an approximation of the Mean Squared Error of Prediction (MSEP).

$$C_p = \frac{SSR}{S^2} + 2p - n \qquad (10)$$

where

SSR is the Regression Sum of Squares

S^2 is the estimate for the variance

p is the number of parameters

n is the number of data points

Theoretically the value of C_p is p. Therefore, when C_p is approximately equal to p, the model is good. Draper and Smith suggest using this criterion in conjunction with

stepwise regression to obtain the best subset (Draper, 1981:341). It should be noted, however, as the variance approaches zero, the C_p statistic can not be calculated. Therefore this method has limitations especially when the fit is perfect.

Barr pointed out a weakness of Mallows C_p. Since S^2 in the C_p statistic is estimated from the original variable pool, it could be biased and larger than the true variance (Barr:5). If this is the case, the C_p statistic will be deflated causing the wrong model to be selected.

A limitation of C_p, as well as many other statistics, is that it "depend[s] on the observed data only through sufficient statistics, so they model average behavior of the fit of a model to the data" (Weisberg, 1981:27). Weisberg developed a procedure which allocates the C_p statistic to individual cases. The advantage of Weisberg's procedure is if the model under consideration is biased, it provides a means to determine the bias of using a subset model instead of the entire model (Weisberg, 1981:28).

Another application of the C_p statistic is to choose the model which has the smallest C_p value. (Judge, 1985:863) By choosing the model with the minimum C_p, it is believed that one is choosing the model with the minimum prediction error. This is appealing, especially when it is difficult to determine the optimal subset using the C_p close to p crite-

rion. Since the Min C_p criterion is based on minimum prediction error it is based on a sound principle. However, like the C_p-close-to-p criterion, Min C_p is derived under the assumption that the independent variables are fixed. Since this rarely happens in practice, there is some question to the usefulness of the Min C_p criterion. Judge, Griffiths, Carter, Lutkepohl, and Lee recommend that the Min C_p procedure not be used in any applied work (Judge, 1985: 864)

Coefficient of Determination. The coefficient of determination, R^2, is a statistic which gives an estimation of the amount of variation about the mean which is explained by the model.

$$R^2 = \frac{\sum_j (\hat{Y}_j - \bar{Y})^2}{\sum_j (Y_j - \bar{Y})^2} \qquad (11)$$

where

$\hat{Y}_j$ is the predicted value of Y_j.

Y_j is the actual value of Y_j

$\bar{Y}$ is the mean of Y.

At first one might believe that it is desirable to find the model which has the maximum R^2, since it explains the most

variation about the mean. However, this is not necessarily the best. Certainly when we look at the R^2 value we would like to see a large value, but it should not be used as the only measure for subset selection. Maximum R^2 receives little praise as far as its usefulness in determining a good fit. The major pitfall of using R^2 is that whenever a variable is added, it will increase R^2. R^2 will increase regardless of whether the variable has anything to do with the dependent variable. According to Healy 1986, "In particular, the multiple correlation coefficient is not really a regression-related concept at all. It is basically defined to be the largest possible correlation between the y-variate and any linear function of the x's and this only makes sense when y and x's have a joint probability distribution" (Healy, 1984:608). If maximum R^2 is used as the selection criterion, the model containing all variables will always be selected.

Maximum Adjusted R^2 or Minimum MSE. For simplicity only Maximum Adjusted R^2 will be discussed. However, Maximum Adjusted R^2 and Minimum MSE test exactly the same thing. Adjusted R^2 is related to R^2, but an adjustment has been made for the degrees of freedom. The following equation shows the relationship between R^2 and Adjusted R^2.

$$Adjusted\ R^2 = 1 - \frac{(1-R^2)\ (n-1)}{(n-p)} \qquad (14)$$

According to Draper and Smith, the adjusted R^2 statistic can be used not only to compare models for the same data set (the same variable selection discussed in all other sections of this literature review), but also to compare models taken from two entirely different data sets (Draper, 1981:92). However, they do not recommend using the Adjusted R^2 statistic in the latter role. The Adjusted R^2 statistic (or the minimum MSE criterion) is still widely used in practice.

PRESS$_p$ or S_p. The S_p criterion, originally proposed by Hocking in 1976 (Hocking, 1976:20), has considerable appeal and consequently receives praise in recent years. The S_p statistic is an approximation of the MSEP based solely on the data and number of variables. As is the case with MSEP, the goal of this criterion is to find the minimum value.

$$S_p = \frac{SSE}{(n-p)\ (n-p-2)} \qquad (15)$$

Breiman and Freedman point out that the S_p statistic does not necessary provide an accurate approximation of MSEP, but works nonetheless (Breiman, 1983:132).

The advantages of this method are numerous. Looking at the S_p equation gives the reader an idea of the relative ease with which S_p is calculated. What makes S_p even more appealing is it is based on MSEP. As Thompson points out, "This method [S_p] is based on a sound criterion -- that of minimizing the expected squared distance between the true and predicted values of the dependent variable, Y" (Thompson, 1978:6). Since S_p is an approximation of MSEP, it can be used like MSEP to determine the optimal number of regressors to include in the model (Breiman, 1983:132).

S_p is not without its disadvantages. It must be calculated for all 2k-1 possible subsets (Thompson, 1978:6). Even though it requires relatively little computational effort, it does require that many regressions be run. Through counter examples Brieman and Freedman show that when true variance due to prediction equals zero, the S_p criterion fails to pick the optimal number of variables to include in the model (Breiman, 1983:132).

Stepwise Regression. A more efficient technique, called Stepwise regression, does not consider all the possible combinations of predictors, but selects only the significant predictors and brings them into the model one at a

time. Stepwise regression exists in three versions: 1) Forward Selection, 2) Backward Elimination, or 3) a combination of 1) and 2). Forward Selection starts with no predictors in the model. It then adds significant predictors to the model one at a time. At each iteration, every predictor not yet in the model is tested for significance with respect to the current model, adding the most significant one to the model. The process continues until all predictors improving the fit of the model are included in the model (Thompson, 1987:10). At no point are variables ever taken out of the model. Backward Elimination, the reverse of Forward Selection, starts with every known predictor already in the model. At each iteration, all the insignificant predictors are identified with respect to the current model, and the least significant predictor is eliminated. This process continues until tests indicate that all insignificant predictors, with respect to the current model, have been eliminated. At no point are variables added back into the model (Thompson, 1987:10-11). The combination of both techniques proceeds like Forward Selection except that Backward Elimination is implemented at each step. Each predictor is tested for significance with respect to the current model, and the most significant predictor is added to the model. Each time a new predictor is brought in, every predictor in the new model is tested with respect to the new model to make sure that it is still significant after the addition of

the newest predictor. Predictors in the model are ranked by their significance and the least significant predictor is eliminated. This process continues until all significant predictors are included in the model and all insignificant predictors are eliminated (Thompson, 1987:11).

The overriding question, then, is how does one measure significance among predictors? The most common measure of significance, called the F-statistic, is a ratio that shows how much explanatory power a predictor brings to the model under consideration. To use an F-statistic in Forward Selection stepwise regression, however, one must decide what numerical threshold of the F-statistic is appropriate. Selecting a small threshold F-value may inadvertently admit random predictors into the model while choosing a large F-statistic may cause significant predictors to be omitted.

Miller's Method. Dr. Alan J. Miller suggests an alternate subset selection method -- one which he theorizes could guard against bringing random predictors into the model. He proposes augmenting the set of predictors with an equal number of "dummy" predictors whose values are random numbers. The method then applies Forward Selection stepwise regression and proceeds, according to Miller, until the first known random predictor is selected for inclusion in the model. One then stops the Forward Selection stepwise regression and discards the current model which includes this known random predictor and uses the previous model

(Miller, 1984:395). Any predictors not selected must, therefore, have less significance than the random predictor that Forward Selection attempted to select. Thus, all predictors not selected should be discarded as insignificant (Hocking, 1983:220). Just how well this subset selection method performs on data plagued with collinearity and other problems is one of the questions which inspired this research effort.

IV. Methodology and Model Development

Objective

The goal of this thesis is to gain a better understanding of the Minimum MSE, Minimum S_p, and Minimum C_p variable selection criteria as well as introducing and studying yet a fourth selection criteria: Miller's method. The four techniques will be compared.

Justification

In this research effort, four variable selection techniques were examined: Minimum MSE, Minimum S_p, Minimum C_p, and Miller's method. These methods were chosen for the following reasons:

(1) Ross Hansen's 1988 thesis research had already studied and compared minimum MSE, minimum S_p, and minimum C_p variable selection techniques. The methodology and systematic approach he developed defined and guided this research effort. However, due to recently discovered computer errors in his data sets and analysis programs, much of Hansen's original computations have been re-worked.

(2) Each of these techniques lend themselves to computer implementation, allowing the researcher to conduct useful experiments and gain credible results with a reason-

able amount of computational effort. This is possible because each of these techniques involve absolute criterion. In other words, all four methods can be executed by a series of predetermined decisions. For the first three methods, the MSE, S_p, and C_p statistics for each data set of variables can be calculated by the SAS (SAS, 1985:956) all-subsets regression procedure, R-Squared. The model selected by the Minimum MSE, Minimum S_p, or Minimum C_p methods is simply the one with the smallest value of MSE, S_p, or C_p, respectively. Similarly, for Miller's method, a model for each data set, augmented with the appropriate number of random predictors, can be automatically selected using the SAS Stepwise procedure with the forward selection option. Miller's model is the largest subset of predictors from the associated SAS model such that each predictor is added in the order of significance determined by the associated SAS model and no random predictors are admitted. Upon encountering the first random predictor, the selection process terminates and the current model becomes the model for that data set.

(3) The first three techniques are very powerful, as Hansen points out:

> The first three techniques appear in the last decade's literature. The Minimum MSE procedure used to be one of the most widely used methods. Its appeal over techniques such as Max R^2 stems from its adjustment for degrees of freedom. More recently, S_p seems to have become the most popular technique. Its appeal is based on the principal

> of minimizing mean square errors of prediction (MSEP). The C_p criterion is also based on MSEP, and some authors praise this criterion. (Hansen, 1988:31)

(4) A formal study of the fourth technique, Miller's method, has not been reported in statistical literature to date. Comparing this virtually unknown subset selection technique with the three well-understood techniques, Minimum MSE, Minimum S_p, and Minimum C_p methods, yielded valuable insight into all four methods.

Limitations

Since this thesis extensively employs least squares regression, the results and conclusions are valid only if certain assumptions can be made about the data. As outlined in Chapter 2, the data must be assumed to be representative of the population. Likewise, the error terms must be assumed to be independent and identically distributed from a normal population with an expected value of zero and a constant variance σ^2. Finally, each predictor must be assumed related to the response (Hansen, 1988:32).

Overview

The methodology and approach exercised in this thesis will be similar in content to that used by Ross Hansen in his 1988 study. Only a slight expansion in methodology occurs with the additional implementation of Miller's subset

selection method. This research effort can be divided into roughly four areas of focus:

(1) Data generation.

(2) Model selection.

(3) Generation and analysis of a performance measure for percentage of correct variables.

(4) Generation and analysis of a performance measure for method accuracy.

The data used in this study is the same as that employed by Hansen, except that certain computer errors have been corrected. The data sets contain various known and verifiable statistical properties.

A model was selected from each data set using each of the four variable selection methods. To accomplish this, preliminary models were formed using SAS all-possible subsets and stepwise regression routines. FORTRAN routines then performed the final model selection process for each method on each data set.

Two different sets of performance measures were calculated. The first set, designated PM, was used to evaluate what effect the various statistical properties of the data have on the percentage of correct variables selected in a given model. Response Surface methodology (RSM) and Box and Whisker plots were applied to determine what impact specific statistical properties of the data and the subset selection technique used have on the percentage correct variables

selected for a given group of models (Hansen, 1988:32). The second set, designated TMSEP for Theoretical Minimum Mean Squared Error of Prediction, is used to compare the accuracy of one subset selection technique to another. This is accomplished by comparing models created under different selection techniques to the theoretical model from which the data was originally generated. Box and Whisker plots were also generated to analyze the impact of each factor on the accuracy of the models a method selects.

Data Generation

Since this study compares its results with the results of Hansen's study, part of the data used came directly from Hansen's study. The Hansen data, however, was augmented with an equal number of random predictors to accommodate Miller's method.

The data for this study was generated from the following equation:

$$Y_i = X_{1i} + X_{2i} + X_{3i} + X_{4i} + \epsilon_i \qquad (16)$$

where Y_i is the response variable.

$X_{1i},\ldots,X_{4i}$ are randomly generated predictors.

ϵ_i is a noise term to create variance in the model.

Most simulation studies investigate subset selection techniques with all significant predictors plus some unknown random variables included among the group of predictors from

which the model is created. This study attempted to find what happens when one of the significant predictors is deleted entirely from consideration. After the data is created by equation (1), the X_4 predictor is dropped from consideration. This simulated the situation which arises when a significant predictor is unknown and not considered. Additionally, either one or three noise variables were included in the predictor pool to simulate data collected on predictors thought to be significant but, in reality, extraneous (Hansen, 1988:34-35). Furthermore, when Miller's variable selection method was implemented, the predictor pool (consisting of both significant and extraneous predictors) was doubled in size by the addition of an equal number of known random predictors. The number of random predictors added always equaled the number variables already in the predictor pool. In practice, however, the actual data sets were not permanently expanded. SAS allowed each data set to be temporary expanded while running a stepwise analysis and implementing Miller's method on each data set.

Factors. To understand how the various statistical properties of the data effect each of the four techniques studied, six potentially significant statistical properties or factors were chosen a priori and the data sets were generated based on these six factors. RSM was used to construct an equation made up of significant factors and factor interactions which adequately predicts the usefulness

of each method (Hansen, 1988:35-36). The six factors considered in this study were:

(1) The number of extraneous variables in the original group of predictors. These variables model predictors which are believed significant, but are actually random, extraneous predictors (denoted by EX_1, EX_2, EX_3). Because these variables are noise, they are theoretically independent of the dependent variable. In this study, at the low setting the number of extraneous variables is 1 and at the high setting, 3.

(2) The amount of correlation among the predictors which are not extraneous, random variables (denoted by X_1, X_2, X_3, X_4). At the low setting the variables are orthogonal, or have zero correlation, while at the high setting they are highly correlated with a correlation of 0.9.

(3) The variance of the extraneous predictors. The low setting for the variance is 1, and the high setting is 100.

(4) The variance of the significant predictors. The low setting for the variance is 1, and the high setting is 100.

(5) The sample size. The low setting for sample size is 10, while the high setting is 20. The low setting was set by Ross Hansen in his study of the S_p criteria -- any smaller and S_p could not be calculated. Hansen's bounds on sample size were adopted to facilitate method comparison (Hansen, 1988:35-36).

(6) The variance of the error term. The low setting for the variance of the error term is 0.0625, and the high setting is 0.25.

Data Sets. Sixty data sets were generated for each of the 64 high/low combinations of the six factor settings. In the literature, each combination of factor settings is typically referred to as a design point in the experiment. In this case, the experiment was to determine what effect each of the six factors has on PM. Hansen wrote automated routines which created each group of the sixty data sets at

each of the sixty-four design points and put them in a file related to the design point. For this thesis effort these files were renamed 01.dat, 02.dat,..., 64.dat). In all, 3840 data sets were generated. Appendix H contains FORTRAN code which was used to verify that the data sets do indeed possess appropriate statistical properties. A close examination of Hansen's data revealed that he used the "natural order" for generating all-possible combinations of the factor settings. To accomplish this, he first established a permanent factor order for future reference.

Table 1.
Factor Order for Data Generation

Order	Factor Description	Values		Factor Symbol
		Low	High	
1	# of extraneous predictors	1.0	3.0	A
2	Correlation among indep. predictors	0.0	0.9	B
3	Variance of ext. predictors	1.0	100.0	C
4	Variance of indep. predictors	1.0	100.00	D
5	Sample Size	10.0	20.0	E
6	Variance of the error term	0.0625	0.25	F

The factors were then varied according to the "natural order". Factor A is varied most rapidly from its low to high

setting, followed by Factor B, C, D, E, and F. The following table gives a example of factor combinations at several design points. For the sake of brevity, "A" means factor A at its high setting and "a" means factor A at its low setting and so forth.

Table 2.
Mapping of Design Points to Factor Settings

Design Point	Data File	Factor Settings
1	01.dat	a b c d e f
2	02.dat	A b c d e f
3	03.dat	a B c d e f
4	04.dat	A B c d e f
5	05.dat	a b C d e f
6	06.dat	A b C d e f
7	07.dat	a B C d e f
8	08.dat	A B C d e f
9	09.dat	a b c D e f
. . .	. . .	. . .
64	64.dat	A B C D E F

Generating the data in this systematic fashion results in an equation relating the performance measure for each subset selection method to these six factors. Before the performance measures can be generated, however, models must be selected using each technique on each data set.

Model Selection

Generally speaking, the variable selection process for all four methods involved employing SAS routines to develop a set of models and then filtering through those models with FORTRAN programs, selecting a model by each method. The implementation of this methodology was similar for the Minimum MSE, Minimum S_p, and Minimum C_p variable selection techniques, but differed for Miller's method. Appendices A and B clearly outline these techniques and reveals these differences.

The reader should keep in mind that the best model at each design point consist of only three predictors: X_1, X_2, X_3 because X_4 had been discarded after data generation. Extraneous predictors, EX_1, or EX_1, EX_2, and EX_3, were added to create the experiment. Although the experimenter knew these were extraneous variables and that they should not be selected for inclusion in the model, the three significant predictors and the extraneous predictor(s) were presented nevertheless to the selection process as legitimate predictors.

Minimum MSE, Minimum S_p, and Minimum C_p Methods. Separate processing was performed for data sets possessing one extraneous predictor and those with three extraneous predictors (see Appendix A). The all-possible subsets SAS routine, RSquared, was used to generate the models. Fifteen models were generated for design points with 4 variables in

the pool and 63 models for design points with 6 variables in the pool and the MSE, C_p, and S_p statistics calculated for each model. The two different quantities of models are due to the number of predictors (p) being considered and is equal to 2^p-1. FORTRAN programs then filtered through the models for each data set and selected three models for each data set: one with the smallest MSE statistic, one with the smallest C_p statistic, and one with the smallest S_p statistic.

Miller's Method. Again, it was necessary to handle the processing separately for data sets possessing one extraneous predictor and those with three extraneous predictors (see Appendix B). To employ Miller's method, the data sets were purposely augmented with an equal number of known random predictors. Depending on whether the number of extraneous variables in the data set is 1 or 3, either 4 or 6 random predictors, respectively, were added to the data set. Miller's method effectively doubled the number of predictors in the pool at each design point. The total number of predictors under consideration by Miller's selection process at each design point varied from 8 (3 unknown true predictors, 1 unknown random predictor, 4 known random predictors) to 12 (3 unknown true predictors, 3 unknown random predictors, 6 known random predictors).

Once augmented, the SAS Stepwise routine using Forward Selection processed each data set. One should note that the

F-to-enter threshold criteria was set to 1 to assure that at least one of the known random predictors would be admitted to the model. Once a model was generated for each data set, FORTRAN programs were used to generate a model for each data set via Miller's method.

Performance Measure (PM) for the Percentage of Correct Variables

Justification. How one rates the performance of a subset selection technique is a critical issue. Adopting a reasonable, logical rating system eventually led to the development of equations which related the success of a method to the statistical properties of the data to which it was applied. Hansen contends that there are no guaranteed methods to screen out extraneous variables (random noise terms which do not contribute at all to the model). Furthermore, he contends that once in the variable pool, there is no criterion which guarantees that no extraneous variables will be chosen for the model. Even the all-subset procedure, which A.J. Miller contends performs quite well, occasionally chooses extraneous variables (Hansen, 1988-:32-33).

Since there really are no "guaranteed methods" for capturing all the true variables, an excellent measure of performance is to rate the success of a subset selection

method by the percentage of variables chosen correctly. This index, referred to as PM, is calculated as follows:

$$PM = \frac{number\ of\ correct\ variables\ chosen}{number\ of\ variables\ chosen} \qquad (17)$$

This study used PM to examine the relative contributions of the six factors as they relate to the performance of each of the four subset selection methods studied. Furthermore, PM is a logical choice for two reasons. First, the best model may not include all the predictors it is generated from, but only the most significant. Even though a response value may have been generated from three predictors, the best model may only contain two of those predictors. Therefore, PM compensates by determining the percentage of correct variables chosen. Second, PM takes in to account the number of extraneous variables chosen. It is worse to select a model with only two predictors, one of which is extraneous, than it is to select a model containing five predictors, one of which is extraneous. PM adjusts accordingly (Hansen, 1988:33).

Calculation of PM. At each of the 64 design points, 60 models were generated. FORTRAN routines examined the 3840 (64 times 60) models produced and selected a model based on the criteria for each method studied. In this final stage of model selection the FORTRAN programs also collected the following statistics at each of the 64 design points:

(1) The total number of predictors chosen in all 60 data sets.

(2) The total number of correct predictors chosen among all 60 data sets (Hansen, 1988:39).

Using these statistics, PM was calculated for each method at each design point. With PM in hand, the experiment was set up and the relationships determined between the PM and the six factors.

Experiment. When using RSM it is convenient to work with coded factors (-1,1 variables) for the following reasons:

(1) By coding the factors, the resulting predictors are of the same magnitude.

(2) Calculations are simplified.

(3) The resulting design matrix, Z, is orthogonal. Consequently, stepwise regression can be used to find the significant factors with confidence (4:36).

In general, translating a variable from uncoded space to coded space is as follows:

$$Z = \frac{X - \frac{HIGH + LOW}{2}}{\frac{HIGH}{2} - \frac{LOW}{2}} \qquad (18)$$

where **X** is the variable in uncoded space
Z is the variable in coded space
HIGH is the upper bound on the uncoded variable

LOW is the lower bound on the uncoded variable

The following equations were necessary to code the variables:

$$Z_1 = \frac{A-2}{1} \tag{19}$$

$$Z_2 = \frac{B-0.45}{0.45} \tag{20}$$

$$Z_3 = \frac{C-50.5}{49.5} \tag{21}$$

$$Z_4 = \frac{D-50.5}{49.5} \tag{22}$$

$$Z_5 = \frac{E-15}{5} \tag{23}$$

$$Z_6 = \frac{F-0.15625}{0.09375} \tag{24}$$

where $Z_1,\ldots,Z_6$ are the coded variables.

Table 3.
Variable Coding for Response Surface Methodology

Variable Description	Uncoded Variable Name	Non-Coded		Coded Variable Name	Coded	
		Low	High		Low	High
Number of ext. vars.	A	1.0	3.0	Z_1	-1	1
Correlation of ind. vars.	B	0.0	0.9	Z_2	-1	1
Variance of ext. vars.	C	1.0	100	Z_3	-1	1
Variance of ind. vars.	D	1.0	100	Z_4	-1	1
Sample Size	E	10.0	20.0	Z_5	-1	1
Variance of error term	F	0.0625	0.25	Z_6	-1	1

It seems reasonable to assume significance of individual factors as well as the significance of interactions between factors. To insure that estimates for both these main factors and their interactions can be accurately calculated, a full 2^6 factorial design is necessary. To construct the design matrix for a full factorial design, the coded factors are varied from their low to high settings with the first coded main factor being varied most rapidly, the second varied next most rapidly, and so forth. The interaction terms are simply the product of the corresponding coded main factors. An example of this process using full 2^3 factorial design is summarized in the table below.

Table 4.
Example of Coding Interaction Variables

Coded Settings	Z_1	Z_2	Z_3	Z_1Z_2	Z_1Z_3	Z_2Z_3	$Z_1Z_2Z_3$
	-1	-1	-1	1	1	1	-1
	1	-1	-1	-1	-1	1	1
	-1	1	-1	-1	1	-1	1
	1	1	-1	1	-1	-1	-1
	-1	-1	1	1	-1	-1	1
	1	-1	1	-1	1	-1	-1
	-1	1	1	-1	-1	1	-1
	1	1	1	1	1	1	1

If a design with less than 2^6 runs is used, information on some of the high order interactions would be unobtainable (Hansen, 1988:38).

Results. The significant factors that contribute to the PM were selected using Stepwise regression with Forward Selection and Backward Elimination (since the design matrix for this experiment was orthogonal). The resulting equations indicate which factor or factor combinations were most significant in increasing the PM, the percentage of correct variables, for a particular method. These equations are not intended to predict the percentage of correct variables, given certain factor settings. The role of these equations, however, is restricted to determining which factors are significant and how they contribute to the percentage of correct variables in a model. On this basis, the four

subset selection methods can be compared. Three similar equations were discovered by Hansen, one for each of the three criteria he studied (Hansen, 1986:39). The analysis which follows is based on the assumption that the closer a PM value is to one, the better a method's performance. Factors which cause PM to become closer to one are desirable.

PM Equations. Using the statistical package STATISTIX version 4.0 a 2^6 full factorial design matrix was created and augmented with the PM vector (STATISTIX, 1992). This design matrix was then exported to the SAS system where a Stepwise regression procedure was run, generating the equations below, as outlined in appendix E (SAS, 1985). An equation was generated for each method studied and shows how that method's performance is related to the factors under which it was applied.

Minimum MSE.

$$\begin{aligned} PM_{MSE} = {} & 0.78 - 0.10(A) + 0.0023(D) + 0.006(E) + 0.0062(F) \\ & + 0.003(AE) + 0.003(AF) - 0.003(DF) - 0.006(EF) \\ & - 0.003(AEF) \end{aligned} \tag{25}$$

Minimum S_p.

$$\begin{aligned} PM_{SP} = {} & 0.85 - 0.07(A) + 0.002(D) + 0.007(E) + 0.007(F) \\ & + 0.002(AD) + 0.006(AE) + 0.003(DE) + 0.007(AF) \\ & - 0.002(DF) - 0.005(EF) - 0.002(ADF) - 0.006(AEF) \end{aligned} \tag{26}$$

Minimum C_p.

$$PM_{CP} = 0.84 - 0.07(A) + 0.003(D) + 0.007(E) + 0.008(F) + 0.002(AD) + 0.006(AE) + 0.003(DE) + 0.008(AF) - 0.003(DF) - 0.005(EF) - 0.002(ADF) - 0.006(AEF) \quad (27)$$

Miller's Method.

$$PM_{MILLER} = 0.88 - 0.04(A) + 0.01(B) + 0.02(E) + 0.01(AB) + 0.008(AE) - 0.008(BE) - 0.008(BCEF) \quad (28)$$

Summary of Effects.

Table 5.
Main Factor Coefficients of Effects by Method for PM

METHOD→→→ / ↓ FACTOR ↓	Minimum MSE	Minimum S_p	Minimum C_p	Miller's Method
A (ext. vars.)	- 0.1	- 0.07	- 0.07	- 0.04
B(ind. corr.)	0	0	0	+ 0.01
C (ext. σ^2)	0	0	0	0
D (ind. σ^2)	+ 0.0023	+ 0.002	+ 0.003	0
E (sam. size)	+ 0.006	+ 0.007	+ 0.007	+ 0.02
F (error σ^2)	+ 0.0062	+ 0.007	+ 0.008	0
Intercept (μ)	+ 0.78	+ 0.85	+0.84	+ 0.88

Table 6.
Main Factor Effects by Method and Rank Order of Significance for PM

METHOD→→→ / FACTOR	Minimum MSE	Minimum S_p	Minimum C_p	Miller's Method
A (ext. vars.)	1st	1st	1st	1st
B (ind. corr.)	No effect	No effect	No effect	2nd
C (ext. σ^2)	No effect	No effect	No effect	No effect
D (ind. σ^2)	4th	4th	4th	No effect
E (sam. size)	3rd	2nd	2nd	3rd
F (error σ^2)	2nd	3rd	3rd	No effect

All Four Methods. The following results pertain to all methods:

(1) The fewer the number of extraneous variables the better the performance.

(2) Larger sample sizes also yielded better performance.

(3) The variance of the independent variable had little effect on the performance of any method.

Minimum MSE, Minimum C_p, Minimum S_p Method. The following additional results were observed for these methods:

(1) Higher variances on the independent variable yielded better results.

(2) Higher variances on the error term give better results.

(3) They were not affected by the correlation of the independent variables.

Miller's Method. The following additional results were observed for this method:

(1) The method did better when the independent variables are highly correlated.

(2) It was not affected by the fluctuating variance on any term (independent or extraneous variables or the error term).

Analysis.

To further assess the impact of each factor (A, B, C, D, E, F) on PM for a given method, STATISTIX version 4.0 was used to produce Box and Whisker plots by indicator grouping (STATISTIX, 1992:96). Each PM value was associated with one of eight values or indicators, dividing it into eight equal indicator groupings. To assign the indicator values, an integer "1" through "4" was assigned to PM values according to the method it measured: 1 for minimum MSE, 2 for minimum S_p, 3 for minimum C_p, and 4 for Miller's method. Next, each number was assigned either a plus or minus sign depending on the factor setting of the factor under consideration, plus for high values and minus for low values. A set of indicator values was developed for each of the six factors studied. The six resulting plots reveal much about the useful-

ness of the four subset selection techniques in choosing the correct variables.

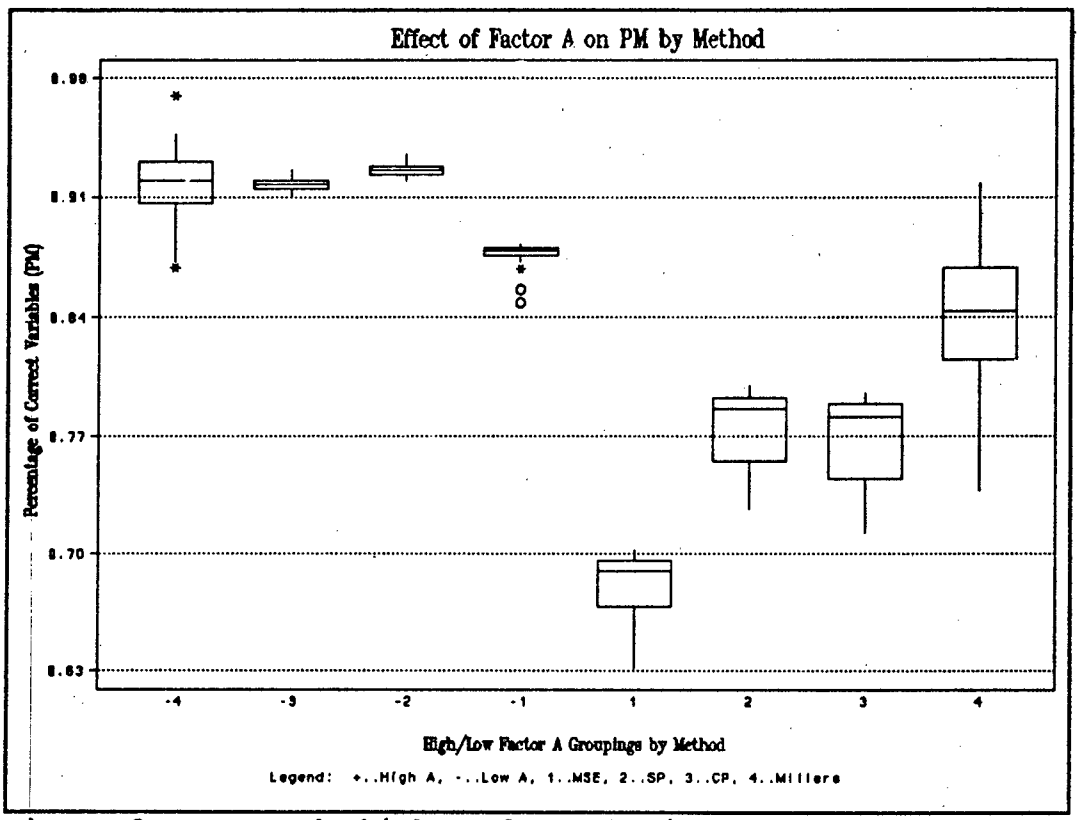


Figure 2. Box and Whisker Plots Showing the Effect of Factor A on PM by Method

Of the four methods applied in selecting correct variables, Minimum MSE is the most affected by the number of extraneous variables in the variable pool. Between the low and high settings, the Minimum MSE method degraded by 15 percent on the average.

On the other hand, Miller's method was a top performer at either setting. Miller's method, on average, outperformed the other three methods being least affected by the number of extraneous variables present. Since in practice, the number of extraneous variables present in a variable pool is not known (by definition), the consistency of Miller's method in dealing with an unknown number of extraneous variables is highly desirable.

When only one extraneous variable is present, the performance of Minimum S_p and Minimum C_p was constant and stable, choosing the correct variable at least 91 times out of 100. Under these circumstances, where few extraneous variables were in the pool, the performances of Minimum S_p and Minimum C_p were predictable and reliable, though not as good as Miller's method.

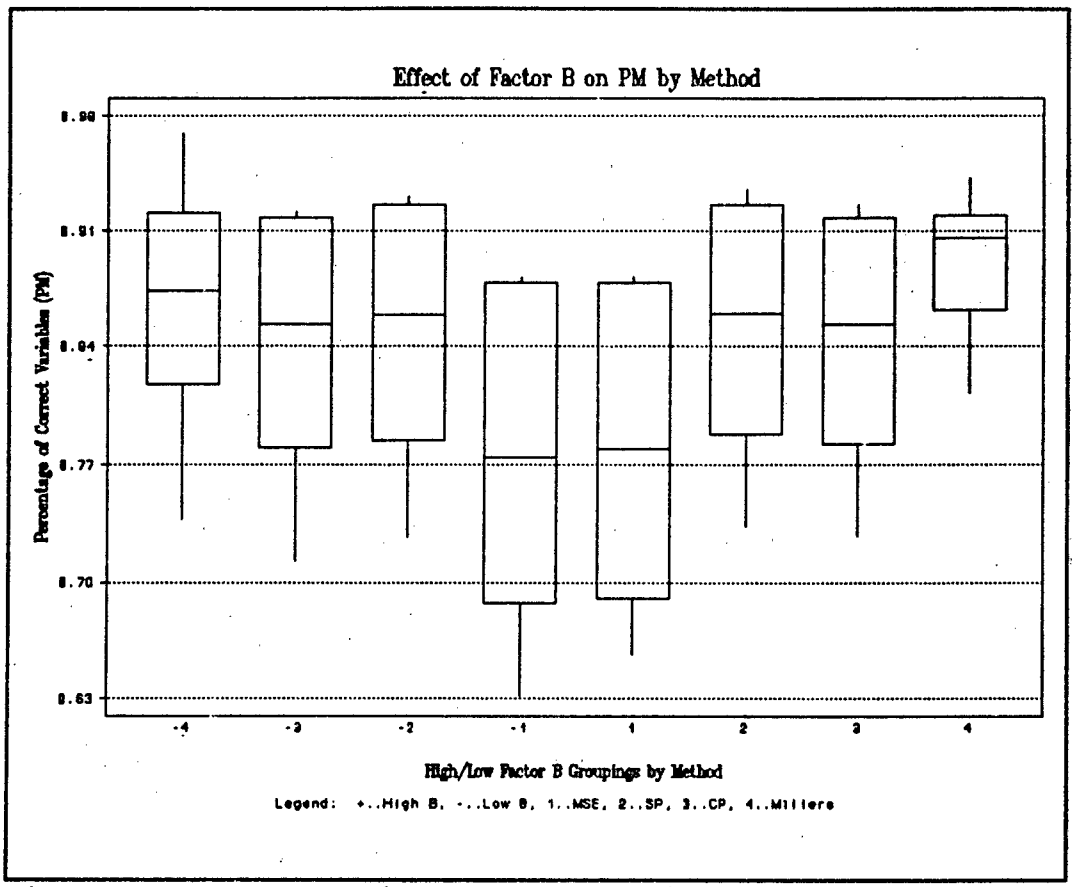


Figure 3. Box and Whisker Plots Showing the Effect of Factor B on PM by Method

Miller's method selects the highest percentage of correct variables at either level and is the only method significantly affected by an increase in correlation among the truly significant predictors. The ability of Miller's method to select correct variables actually increases as the correlation between the correct variables increases. This occurs because the increased correlation among the correct variables causes them to behave as one variable. If any one

correct variable is selected, it is equivalent to all the correct variables being selected.

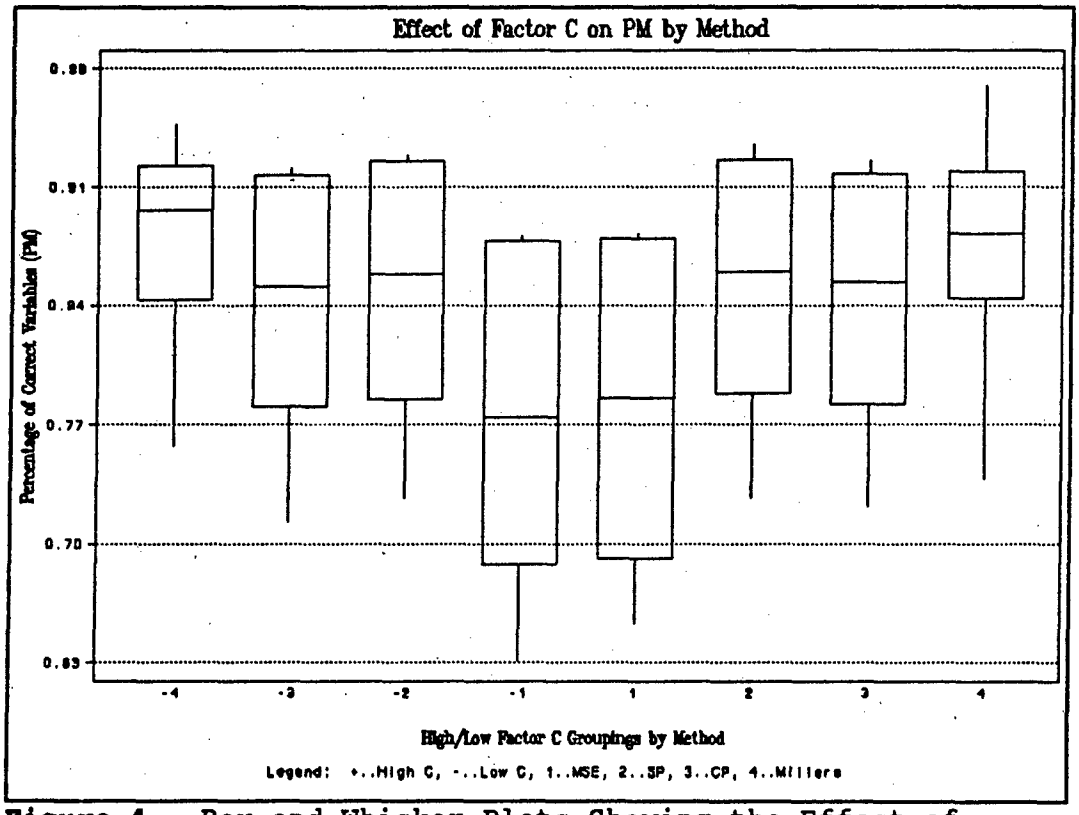


Figure 4. Box and Whisker Plots Showing the Effect of Factor C on PM by Method

Factor C, the variance of the extraneous variable, had little effect on any of the four methods. The median of the MSE method improved slightly with an increase in variance of the extraneous variables while the median of Miller's method decreased slightly.

The Minimum S_p, Minimum C_p, and Minimum MSE methods lag behind Millers method and show a greater variability.

Clearly the Minimum MSE method selects the smallest percentage of correct variables.

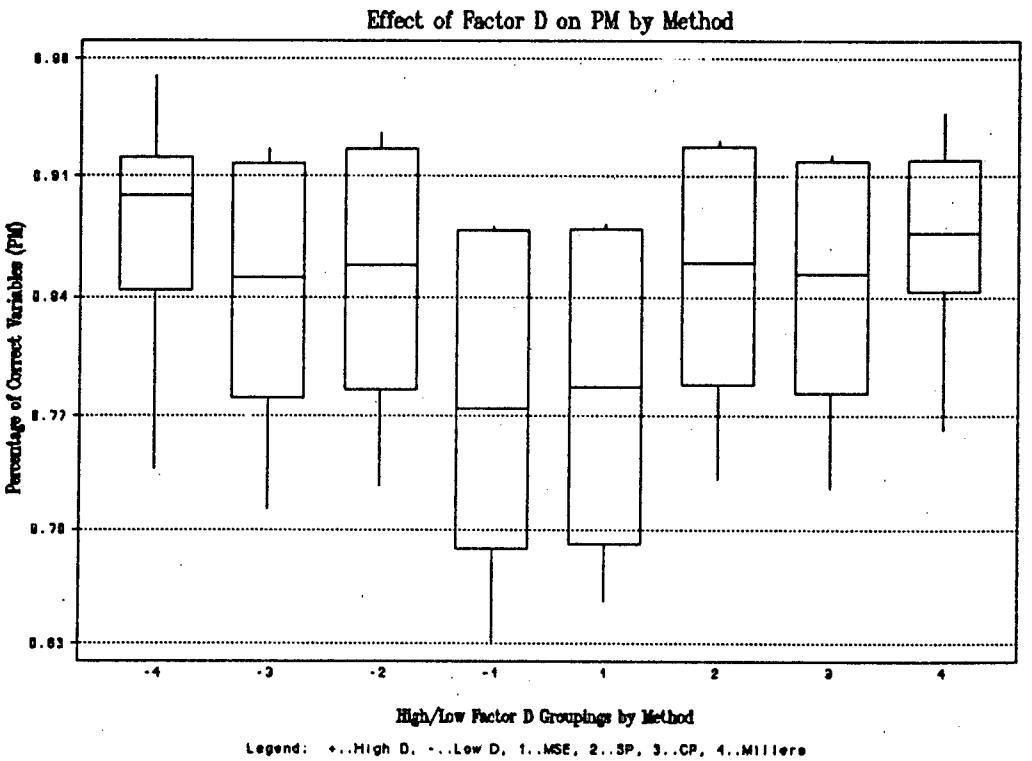


Figure 5. Box and Whisker Plots Showing the Effect of Factor D on PM by Method

Again, this plot for factor D, like that of factor C, shows that the variance of the correct variables has little effect on the percentage of correct variables chosen for any of the four methods.

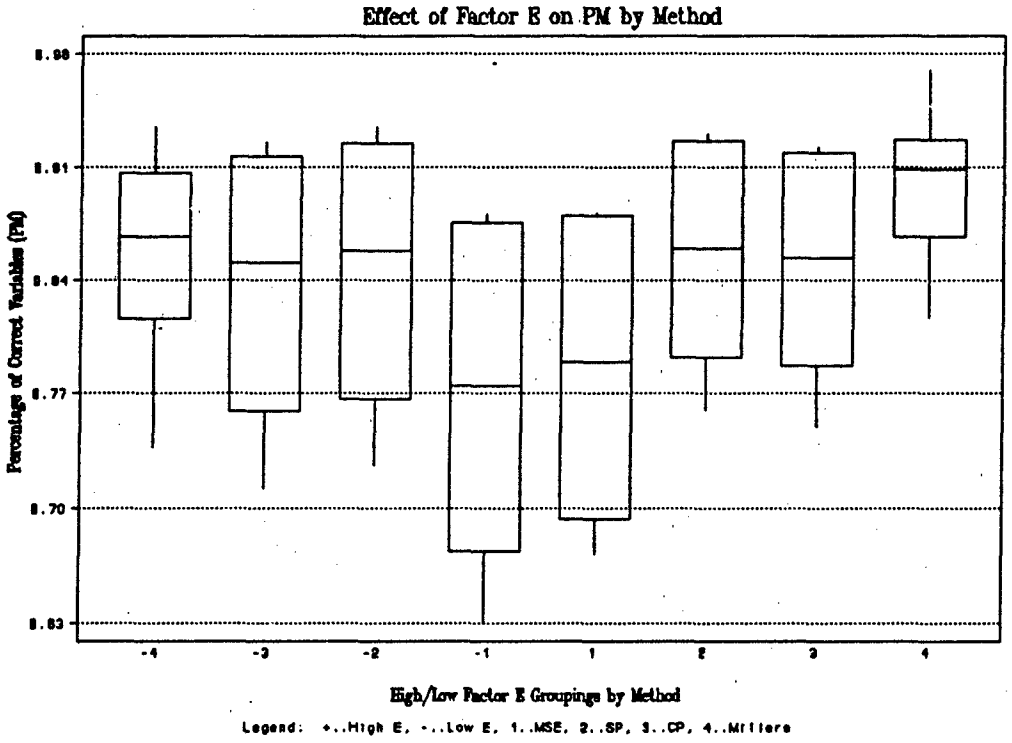


Figure 6. Box and Whisker Plots Showing the Effect of Factor E on PM by Method

Increasing the sample size, factor E, increases the median performance of Miller's method by 5 percent. The Minimum MSE method also improves slightly as sample size increases. The Minimum S_p and Minimum C_p method are not effected.

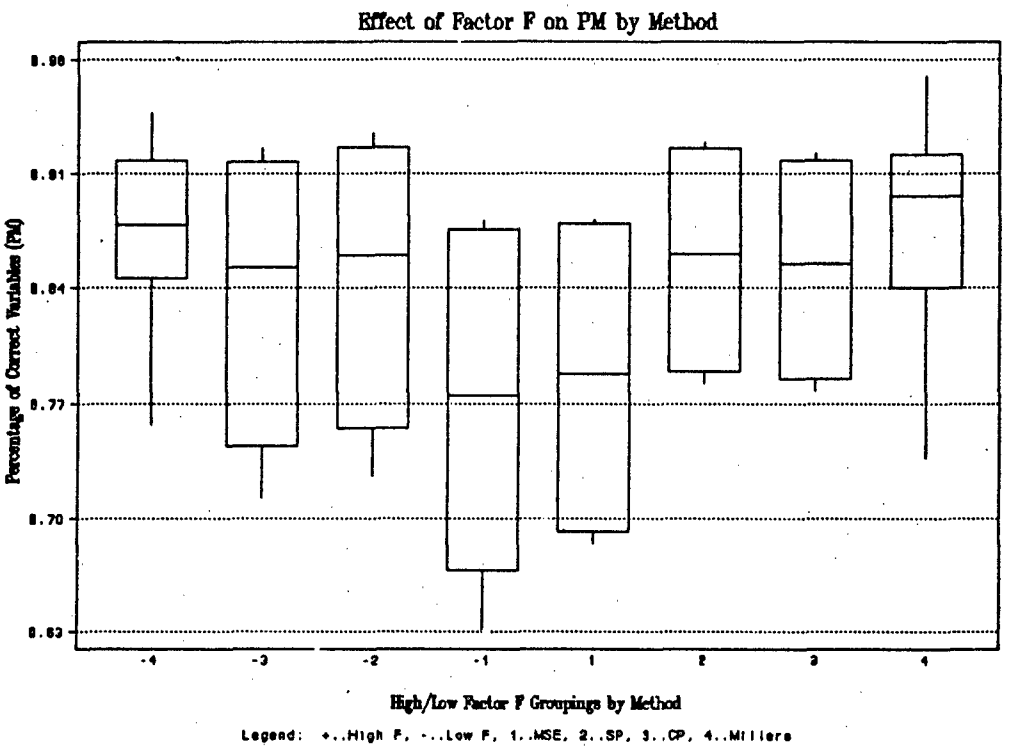


Figure 7. Box and Whisker Plots Showing the Effect of Factor F on PM by Method

Increasing the variance of the error term, factor F, has little effect on any of the four methods. The median performances of the Minimum MSE method and Miller's method is slightly increased at the higher factor levels.

Theoretical Mean Square Error of Prediction (TMSEP) as a Performance Measure of Model Accuracy

Justification. Thus far, analysis has been limited to studying the effects which varying factors have on a method's ability to select the correct variables. PM, however,

when employed as an index for comparing different selection methods, favored techniques which select models with the highest percentage of correct variables. Although models with a high percentage of correct predictors are desirable, methods which select such models may do so by selecting more variables overall. In such models, the ratio of extraneous variables to all the variables may be small, but the absolute number of extraneous variables may be larger than desired simply because of the sheer number of variables selected. Comparing techniques on the basis of PM may favor methods which create these larger models rather than those which create parsimonious models. Therefore, a different, more absolute performance measure was adopted to compare selection techniques in terms of how close the selected models response value is to the true response value. A comparison of how accurately each technique performs can be accomplished using another performance measure known as Theoretical Minimum Mean Square Error of Prediction (TMSEP) and defined by:

$$TMSEP_{k,M} = \frac{\sum_{t=1}^{n_k} (Y_t - \overline{Y}_{k,M})^2}{(n_k - p_{k,M})} \qquad (29)$$

where

$TMSEP_{kM}$ is the TMSEP for data set k using the subset selection technique M

Y_t is the theoretical conditional mean of Y calculated from the underlying data generation model (1) and the data set k.

$\bar{Y}_{kM}$ is the predicted value of Y using the model selected by applying method M to data set k

n_k is the sample size of data set k.

p_{kM} is the number of predictors in the model selected by applying method M to data set k

TMSEP is a good choice for an inter-technique comparison. It compares each method's model at a particular data set to the theoretical model which generated the original data. In theory, TMSEP directly measures how well the predicted model explains the variations in the original data. Furthermore, the TMSEP criterion is a variation of Mean Squared Error Prediction (MSEP), a statistic that has received much praise in the literature. TMSEP and MSEP both calculate the squared difference between the predicted value of Y and the actual value of Y and adjust the value for the degrees of freedom. TMSEP differs from MSEP, however, in its calculation. TMSEP is calculated by squaring the difference between the theoretical Y value (the response from the underlying data generation equation, excluding the error term) and the predicted Y value generated by the model constructed using variable selection procedure, M. The resulting value is the Theoretical MSEP or TMSEP. Since the TMSEP is based on MSEP which has received considerable praise in the literature during the past decade, the TMSEP

is also considered a most promising criterion (Hansen, 1988,43-45).

In defending the credibility of TMSEP, Hansen notes that at first glance, TMSEP appears to unfairly favor the Minimum C_p and Minimum S_p criteria because both are based in minimum MSEP. Furthermore, one might falsely assume that since the S_p criterion and TMSEP are based on the regressors being randomly generated, the TMSEP would favor the Minimum S_p method. Hansen clearly shows this is not the case.

> It is assumed when calculating the S_p and C_p statistics that all relevant variables are included in the variable pool. It is also assumed that the variable pool does not contain extraneous variables. In this study both of these assumptions are violated. Therefore, it is possible that either the MSE, C_p[, or Miller] criterion could outperform the S_p criterion. (Hansen, 1988:46)

Calculating TMSEP. The equation presented thus far to calculate TMSEP does so one data set at a time. Recall that the generated data consists of 64 design points each of which is made up of 60 data sets. In order to compare each of the four variable selection techniques, the TMSEP must somehow be calculated for each technique at each design point. Although generating the TMSEP for each data set is a starting point, a slightly different TMSEP equation is necessary to generate the aggregate TMSEP at each design point.

Starting with the original equation from Hansen's thesis:

$$TMSEP_{k,M} = \frac{\sum_{t=1}^{n_k} (Y_t - \overline{Y}_{k,M})^2}{(n_k - p_{k,M})} \qquad (31)$$

Then, applying algebra yields:

$$\frac{(n_k - p_{k,M})\, TMSEP_{k,M}}{\sigma^2} = \frac{\sum_{t=1}^{n_k} (Y_t - \overline{Y}_{k,M})^2}{\sigma^2} \qquad (32)$$

Hansen assumed a Chi Square distribution (Hansen, 1988:44)

$$\frac{\sum_{t=1}^{n_k} (Y_t - \overline{Y}_{kM})^2}{\sigma^2} \sim \chi^2_{(n_k - p_{kM})} \qquad (33)$$

Then, it follows from equation 31 that:

$$\frac{(n_k - p_{k,M})\, TMSEP_{k,M}}{\sigma^2} \sim \chi^2_{(n_k - p_{k,M})} \qquad (34)$$

Based on the theorem that the sum of independent χ^2 variables is also χ^2, we have:

$$\sum_{k=1}^{60}\left[\frac{(n_k-p_{k,M})\,TMSEP_{k,M}}{\sigma^2}\right] \sim \chi^2_{(n_k-p_{k,M})} \quad OR \quad \chi^2_{\left[\sum_{k=1}^{60} n_k-\sum_{k=1}^{60} p_{k,M}\right]}$$

Now

$$\sum_{k=1}^{60}\left[\frac{(n_k-p_{k,M})\,TMSEP_{k,M}}{\sigma^2}\right] = \sum_{k=1}^{60}\left[\frac{\sum_{t=1}^{n_k}(Y_t-\bar{Y}_{k,M})^2}{\sigma^2}\right] \qquad (36)$$

$$\sum_{k=1}^{60}\left[\frac{(n_k-p_{k,M})\,TMSEP_{k,M}}{\sigma^2}\right] = \frac{\sum_{k=1}^{60}\sum_{t=1}^{n_k}(Y_t-\bar{Y}_{k,M})^2}{\sigma^2} \qquad (37)$$

So:

$$\frac{\sum_{k=1}^{60}\sum_{t=1}^{n_k}(Y_t-\bar{Y}_{k,M})^2}{\sigma^2} \sim \chi^2_{\left[\sum_{k=1}^{60} n_k-\sum_{k=1}^{60} p_{k,M}\right]} \qquad (38)$$

Therefore, the formula for calculating TMSEP is:

$$TMSEP_{D,M} = \sum_{k=1}^{60} TMSEP_{k,M} = \frac{\sum_{k=1}^{60}\sum_{t=1}^{n_k} (Y_t - \bar{Y}_{k,M})^2}{\sum_{k=1}^{60} n_k - \sum_{k=1}^{60} p_{k,M}} \qquad (39)$$

where $TMSEP_{D,M}$ is the TMSEP at design point D using method M

Minimum MSE, Minimum S_p, and Minimum C_p Methods.

Appendix C outlines the data processing to calculate TMSEP for each design point for the above three methods. The processing is similar to that performed at each design point during subset selection for each method. The processing differs in that for each model selected, the coefficients of regression are estimated by SAS. The FORTRAN program uses these coefficients and the original data to generate TMSEP for each design point and each method.

Miller's Method. Appendix D outlines the data processing to calculate TMSEP for each design point for Miller's method. A FORTRAN program creates a SAS program file to calculate the coefficients of regression for each model. This SAS program is executed and the output is filtered and formatted by yet another FORTRAN program. A third FORTRAN program processes this output along with the data sets and calculates the TMSEP for each method and design point.

Experiment. An experiment identical to the one run for PM was run to determine the significant factors for TMSEP. Basically, TMSEP was substituted for PM in the experimental design and then the experiment was run as before, using the SAS Stepwise procedure.

Results. The same comments that applied to PM apply to TMSEP, with one notable exception. Whereas with PM, values $\rightarrow 1^-$ were desirable, with TMSEP values $\rightarrow 0^+$ are the target.

TMSEP Equations. These equations were generated in exactly the same manner as the PM equations

Minimum MSE.

$$\begin{aligned} TMSEP_{MSE} = {} & 22.15 - 1.76(A) - 16.91(B) + 21.6(D) + 3.04(E) \\ & + 1.4(AB) - 1.75(AD) + 0.74(AE) - 16.57(BD) \\ & - 2.4(BE) + 3.03(DE) + 1.38(ABD) - 0.65(APE) \\ & + 0.74(ADE) - 2.36(BDE) - 0.64(ABDE) \end{aligned} \tag{40}$$

Minimum S_p.

$$\begin{aligned} TMSEP_{SP} = {} & 23.22 - 1.54(A) - 17.76(B) + 22.65(D) + 2.29(E) \\ & + 1.27(AB) - 1.52(AD) - 17.4(BD) - 1.8(BE) \\ & + 2.3(DE) + 1.24(ABD) - 1.77(BDE) \end{aligned} \tag{41}$$

Minimum C_p.

$$TMSEP_{CP} = 37.51 - 0.84(A) - 25.72(B) + 36.69(D) + 5.86(E) + 0.79(AB) - 0.82(AD) - 25.19(BD) - 4.08(BE) + 5.77(DE) + 0.77(ABD) - 4.0(BDE) \quad (42)$$

Miller's Method.

$$TMSEP_{MILLER} = 33.02 - 26.22(B) + 32.26(D) - 3.93(E) - 25.69(BD) + 3.32(BE) - 3.84(DE) + 3.27(BDE) \quad (43)$$

Summary of Effects.

Table 7.
Main Factor Coefficients of Effects by Method for TMSEP

METHOD→→→ / ↓ FACTOR ↓	Minimum MSE	Minimum S_p	Minimum C_p	Miller's Method
A (ext. vars.)	- 1.76	- 1.54	- 0.84	0
B(ind. corr.)	- 16.91	- 17.76	- 25.72	- 26.22
C (ext. σ^2)	0	0	0	0
D (ind. σ^2)	+ 21.6	+ 22.65	+ 36.69	+ 32.26
E (sam. size)	+ 3.04	+ 2.29	+ 5.86	- 3.93
F (error σ^2)	0	0	0	0
Intercept (μ)	+ 22.15	+ 23.22	+ 37.51	+ 33.02

Table 8.

Main Factor Effects by Method and Rank Order of Significance for TMSEP

METHOD→→→ / ↓ FACTOR ↓	Minimum MSE	Minimum S_p	Minimum C_p	Miller's Method
A (ext. vars.)	4th	4th	4th	No effect
B (ind. corr.)	2nd	2nd	2nd	2nd
C (ext. σ^2)	No effect	No effect	No effect	No effect
D (ind. σ^2)	1st	1st	1st	1st
E (sam. size)	3rd	3rd	3rd	3rd
F (error σ^2)	No effect	No effect	No effect	No effect

All Four Methods. The following results pertain to all methods:

(1) The higher the correlation among the independent or correct variables, the better the performance.

(2) Lower variances in the independent or correct variables yielded better performance.

Minimum MSE, Minimum C_p, Minimum S_p Method. The following results were additionally observed for these methods:

(1) The higher the number of extraneous variables, the closer the response value is to its true theoretical value.

(2) Smaller sample sizes give better results.

Miller's Method. The following additional results were observed for this method:

(1) Adding extraneous variables causes improvement of the TMSEP for a model.

(2) Better performance is obtained with larger sample sizes.

Analysis. As with PM, Box and Whisker plots were employed to further assess the impact of each factor (A, B, C, D, E, F) on TMSEP for a given method. STATISTIX 4.0 was also used to produce these Box and Whisker plots by forming the indicators in the same manner as before. A set of indicator values are created for each of the six factors studied. The six resulting plots revealed much about the ability of each subset selection technique to create a model close to the actual model.

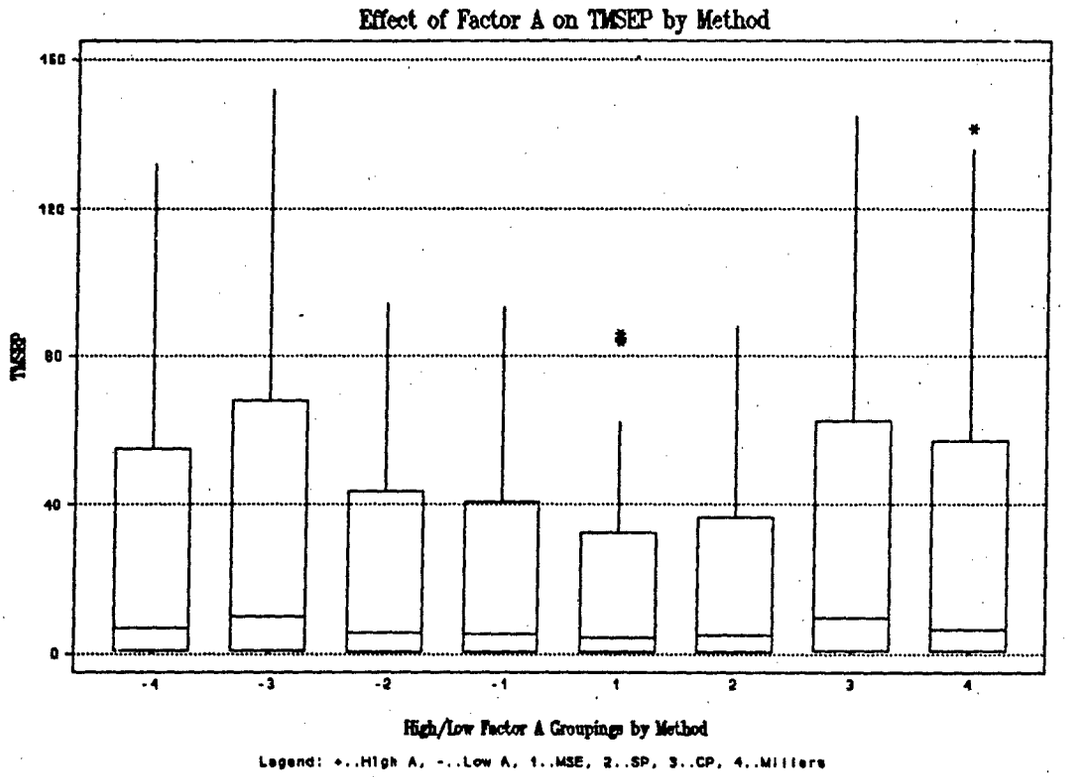


Figure 8. Box and Whisker Plots Showing the Effect of Factor A on TMSEP by Method

The number of extraneous variables involved had very little impact on how close a method came to selecting the absolutely correct model. Of the four methods studied, however, MSE appears to perform best.

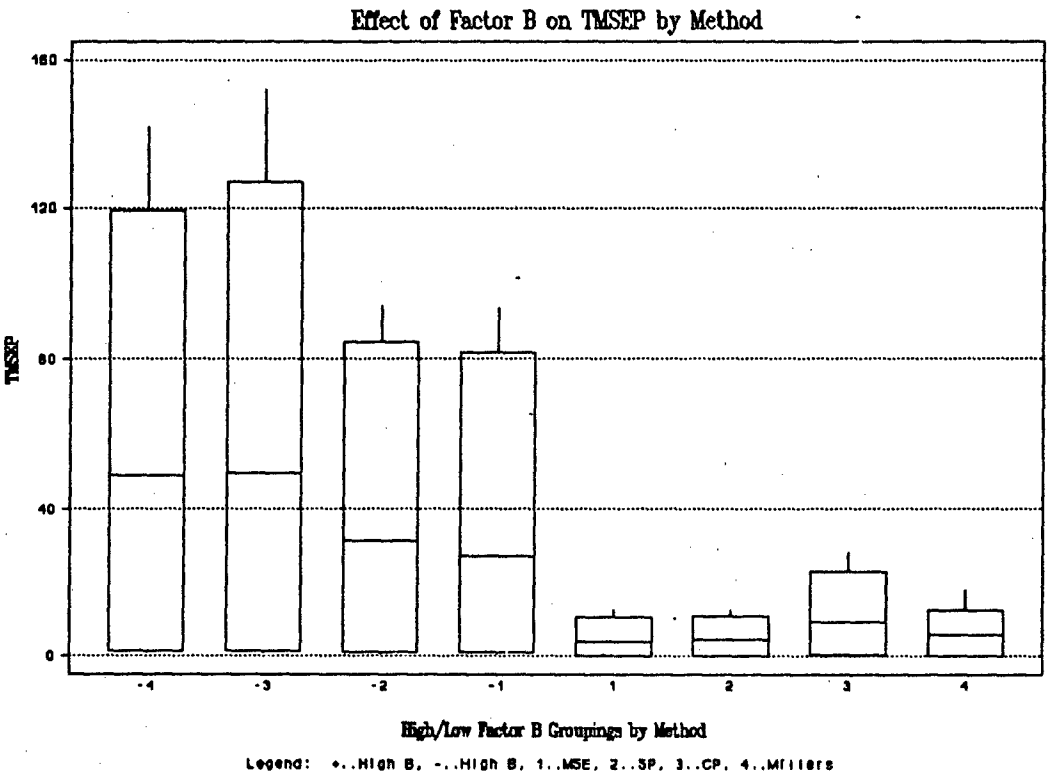


Figure 9. Box and Whisker Plots Showing the Effect of Factor B on TMSEP by Method

Clearly the amount of correlation between the correct variables has a great effect on model accuracy. When the correct variables are highly correlated, one contains almost all of the information contained in all four of them (including the one omitted from the pool).

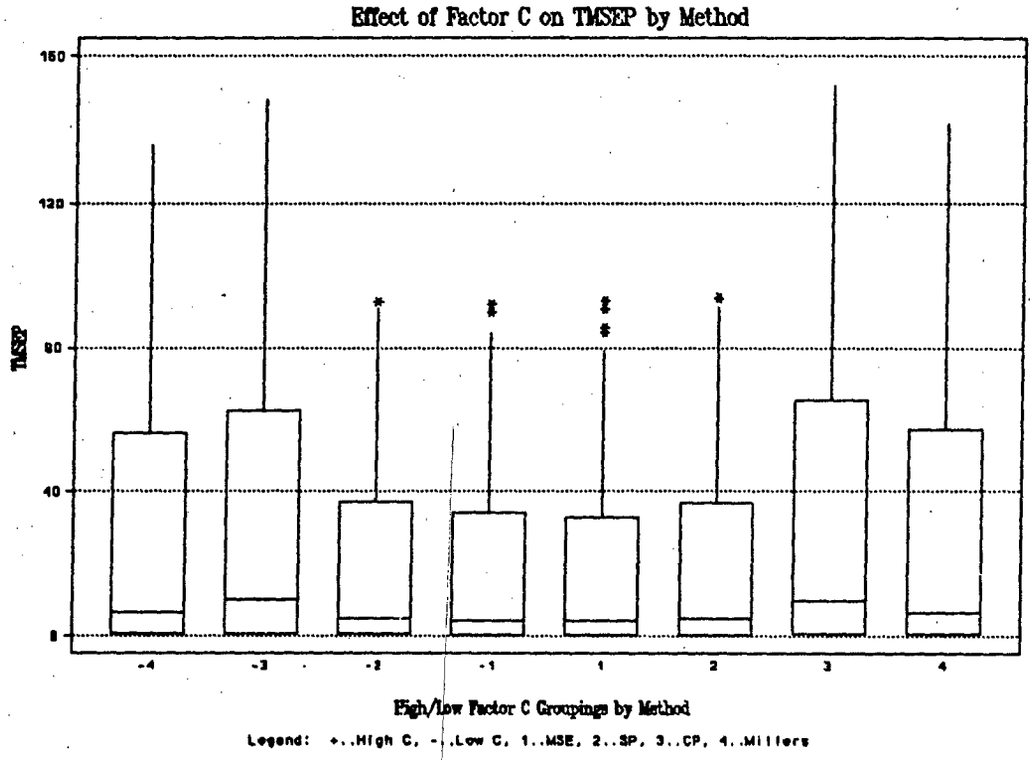


Figure 10. Box and Whisker Plots Showing the Effect of Factor C on TMSEP by Method

Factor C, the variance of the extraneous variables, has no effect on the accuracy of any method. If the focus was on selecting the correct variables, it follows that a change in the extraneous variables would have little effect.

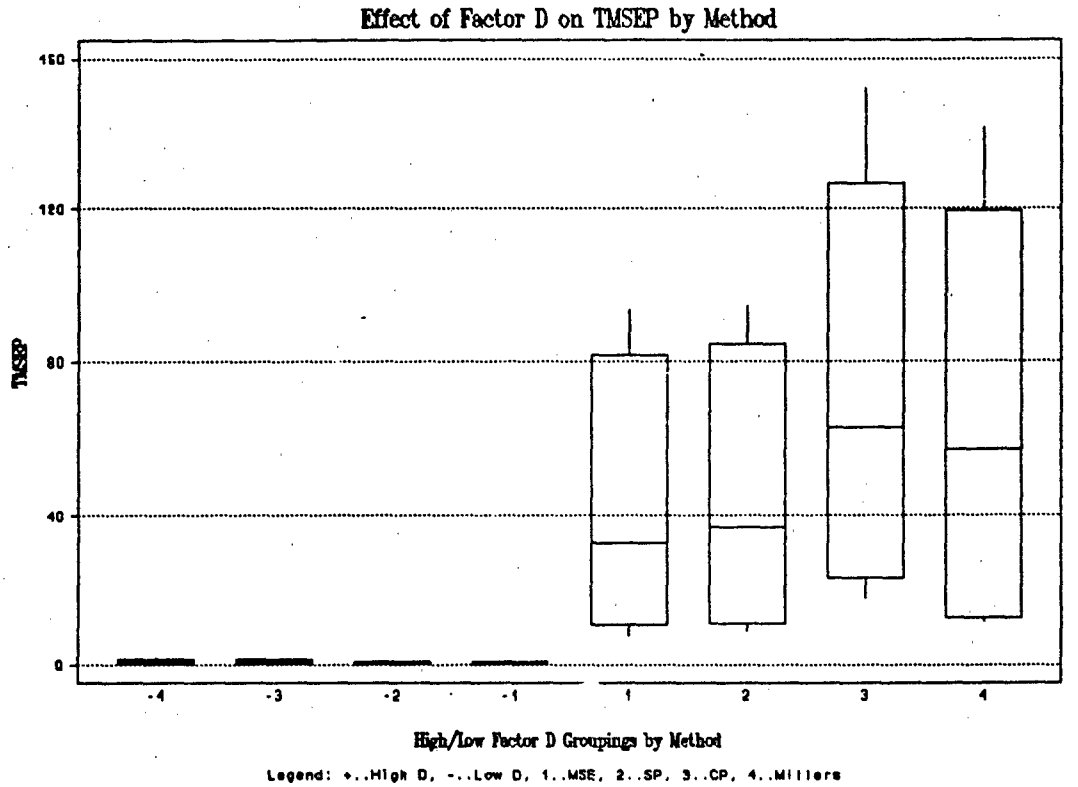


Figure 11. Box and Whisker Plots Showing the Effect of Factor D on TMSEP by Method

The variance of the extraneous variables, factor D, effects the performance of all four methods. Minimum MSE and Minimum S_p methods appear to be more affected than Minimum C_p and Miller's methods.

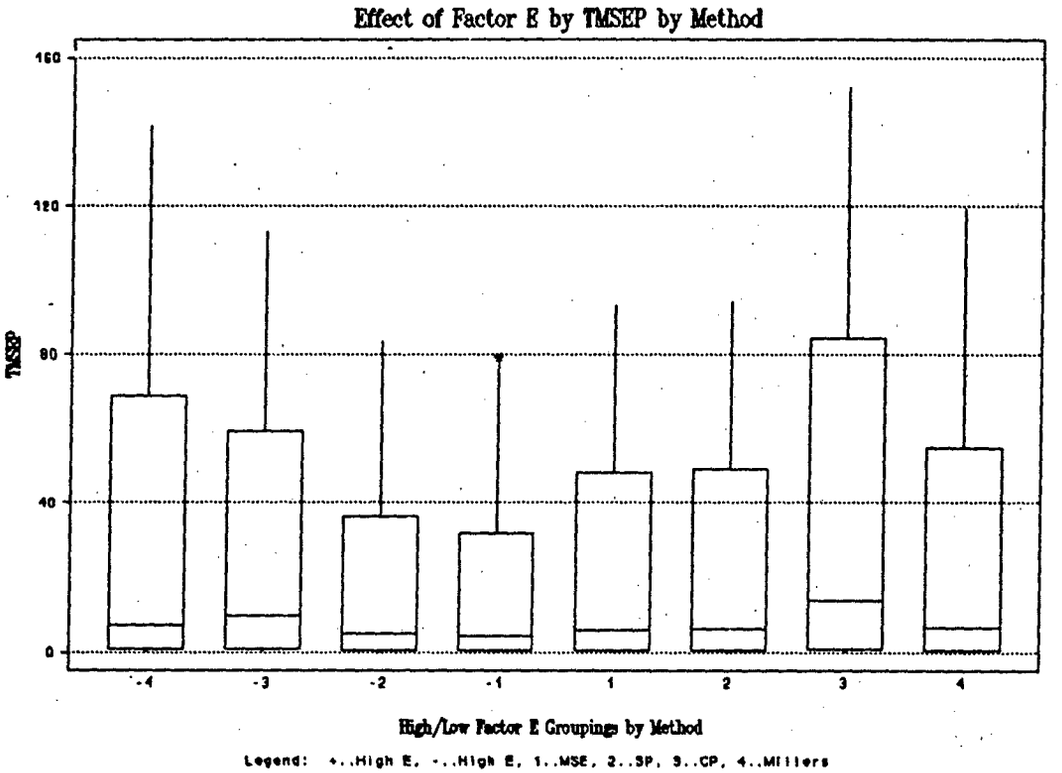


Figure 12. Box and Whisker Plots Showing the Effect of Factor E on TMSEP by Method

Increasing the sample size, factor E, tends to increase the variance in the Minimum MSE, Minimum S_p, and Minimum Cp methods. Miller's method, however, becomes slightly more consistent.

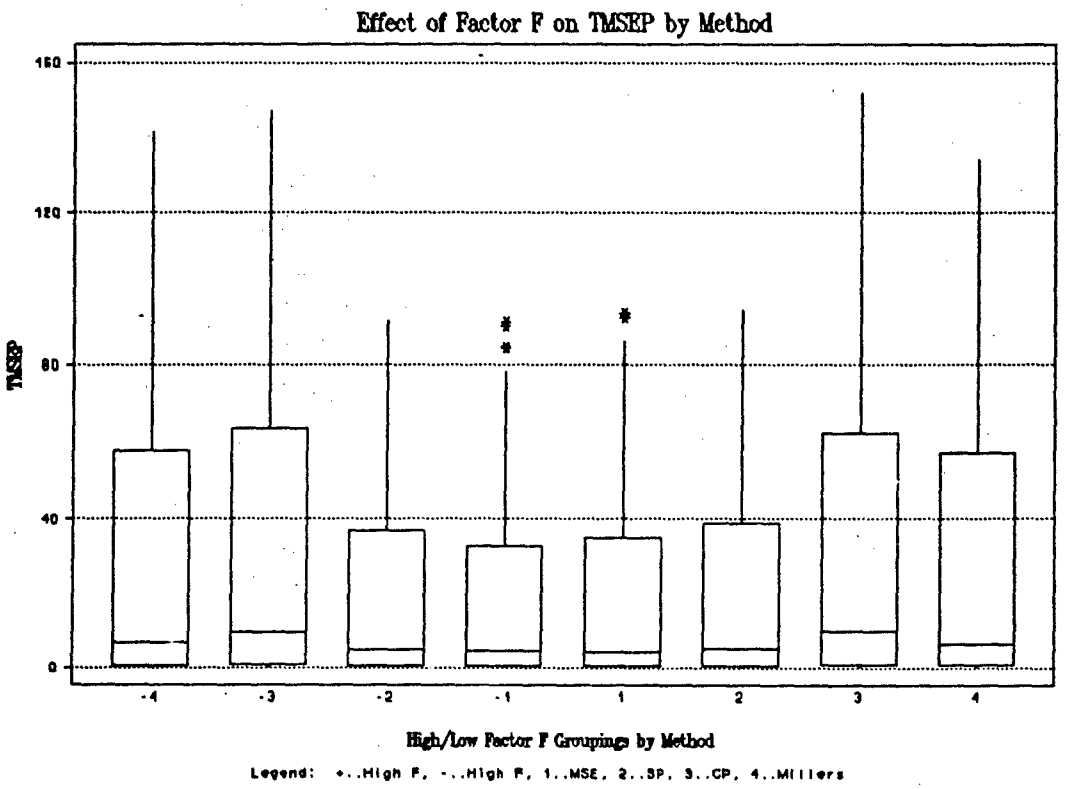


Figure 13. Box and Whisker Plots Showing the Effect of Factor F on TMSEP by Method

The variance of the error term, factor F, has no apparent effect on the accuracy of the four methods studied. All the method were able to filter out the white noise equally well.

V. Conclusions and Recommendations for Further Research

Conclusion

Objective. The objectives of this research were: (1) identify some promising least squares selection procedures discussed in the literature, (2) introduce, implement, and study a variable selection method proposed by Alan J. Miller, and (3) make an extension of Hansen's research by comparing the methods he examined: Minimum MSE, Minimum S_p, and Minimum C_p, with Miller's method.

Techniques Studied. The Minimum MSE, Minimum S_p, and Minimum C_p variable selection techniques have received much praise in the past 20 years. Due to the similarity to the Maximum R^2 criterion and its adjustment for degrees of freedom, Minimum MSE was considered the favored technique fifteen years ago. More recently Minimum S_p and Minimum C_p, both of which are based on MSEP, have received the majority of the praise. Of the two, Minimum S_p is the more practical selection method because it is designed for random regressors (Hansen, 1988:59).

Compared to the three well-known techniques mentioned above, Miller's method was obscure and untested. A literature search revealed only Miller's original reference to the procedure. This research has compared and contrasted Miller's method with the well-accepted techniques, Minimum

MSE, Minimum S_p, and Minimum C_p, and thereby defined its role among current variable screening techniques.

Methodology. To facilitate a comparative analysis of Miller's method and the other methods, Response Surface Methodology was employed with two performance measures. The first, designated PM, measured the percentage of correct variables in a model. The second, Theoretical Mean Squared Error of Prediction (TMSEP), measured the predictive error between the model selected and the theoretical model. A 2^6 full factorial design was setup, yielding the 64 high/low combinations, or design points, of the six factors being studied. Using Hansen's data, which had been generated with 60 replications at each design point, both PM and TMSEP were calculated for each subset selection method at each design point. The SAS Stepwise procedure was used to select significant factors or factor combinations at the $\alpha = 0.01$ level and to generate a linear equation for each combination of performance measure and selection method. Four of these eight equations revealed what each of the six factors and their combinations contributed toward improving the percentage of correct variables (maximizing PM) in a model and the other four examined how the same factors related to minimizing the error between the modeled response and the theoretical response (minimizing TMSEP). STATISTIX 4.0 was then used to produce Box and Whisker plots by performance measure and method. These plots revealed factor effects and provid-

ed a graphical analysis of variance on performance measures by method and factor settings.

Two-Stage Variable Selection Technique. The data used in this thesis attempted to simulate real world data. Extraneous variables were added and one of the significant predictors was totally dropped from consideration after generating the data. In light of these tough, inherent data problems, it was suspected from the beginning of this research effort that a single selection method may not be effective at both screening out the extraneous variables and selecting the final model. Therefore, two performance measures, PM and TMSEP, were examined because they rate selection methods from different vantage points. A selection technique which rated highly under PM would perform well as a screening method prior to final variable selection. During the screening process the objective is to select the greatest number of significant variables (or correct or true variables) while rejecting any extraneous ones. PM measured how well each method accomplished this. On the other hand, a selection technique which rated highly under TMSEP would perform the final variable selection process well. During the final selection process, a set of likely predictors is examined and the final subset selected. One hopes that this final subset of predictors has a response close to that of the theoretically correct set of predictors. TMSEP measured the performance of each method

in this regard. Note that PM and TMSEP were calculable only because the data for this research was generated by a known model. In practice, PM and TMSEP cannot be calculated. It was the intention of this research, therefore, to observe the performance of the four variable selection techniques in question under controlled conditions and to note the conditions under which they perform best.

In a screening situation where PM would apply, all four PM equations and a comparison of their regression factor coefficients indicated that the number of extraneous variables (factor A) was the most significant factor, sometimes by a difference as much as two magnitudes. Box plots of PM for factor A also revealed that Miller's method had the highest median PM value. The equation for $PM_{MILLERS}$ reveals why this occurred. $PM_{MILLERS}$ had the highest intercept value and the number of extraneous vari bles reduced the performance measure by less than half the amount the other PM equations did for the other methods. Obviously, when selecting the independent or correct variables from a variable pool containing extraneous variables, Miller's method was the method least affected by the presence of extraneous variables. Thus Miller's method is the best technique for screening.

Once screened, the variable pool is ready for final model selection. As stated previously, a method's performance during this final selection stage is best gauged by

TMSEP. TMSEP was primarily affected by two factors: the variance of the independent or correct variables (factor D) and the correlation among the same variables (factor B), as the all TMSEP equations reveal. The regression coefficients of factors B, D, and the BD interaction were a magnitude larger than any other coefficients. Closer examination of the TMSEP equations showed that when factor D (variance of the correct variable) was at its low setting, factor B (correlation of the correct variables) caused about the same improvement (decrease) of TMSEP at its high and low levels. When factor D is set high, however, the low setting of factor B worsens (increases) TMSEP and the high settings of factor B improves (decreases) TMSEP. The following analysis graphically depicts this BD interaction using B+D+BD to calculate the weights in each quadrant. This explains the importance of the BD interaction.

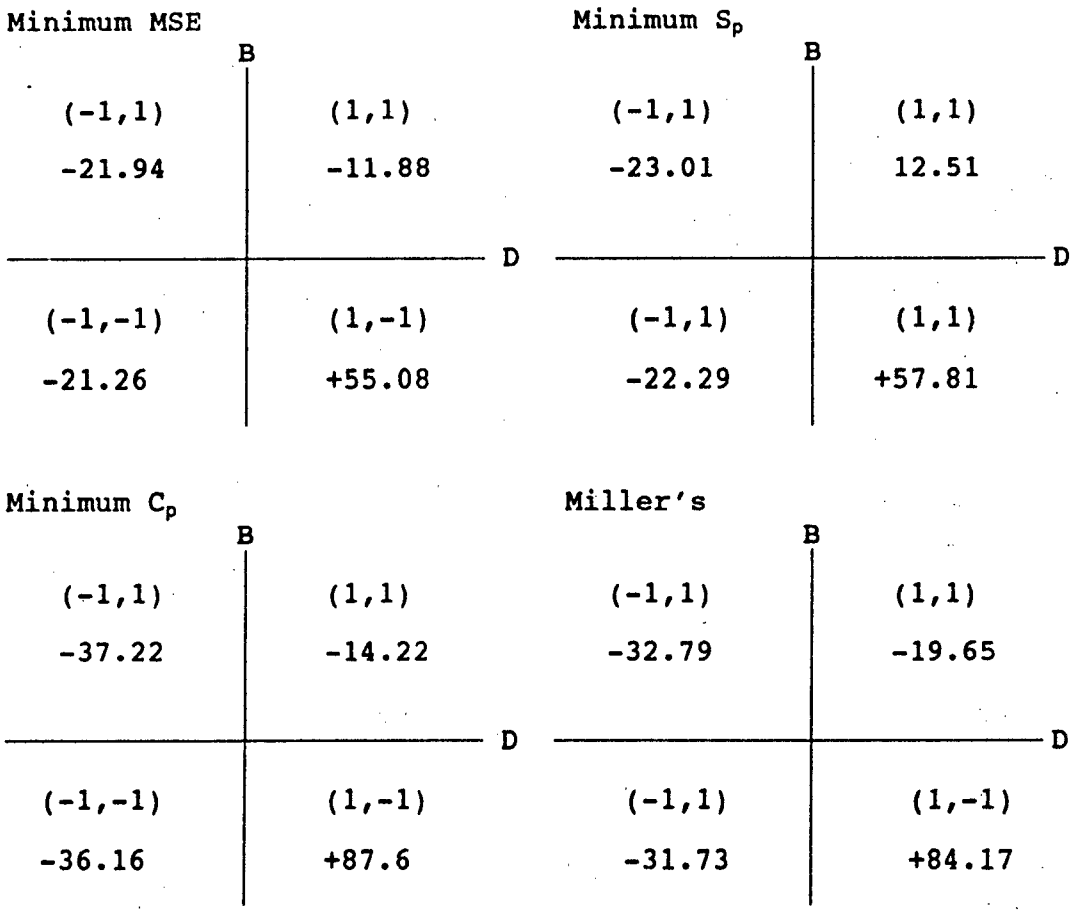


Figure 14. Graphical Analysis of the BD Interaction by Method

Box plots for factors B and D show that the Minimum MSE method had the best median TMSEP, followed closely by the Minimum S_p method. Furthermore, the following box plot for the BD interaction factor confirms that the Minimum MSE method would perform best as a final selection technique.

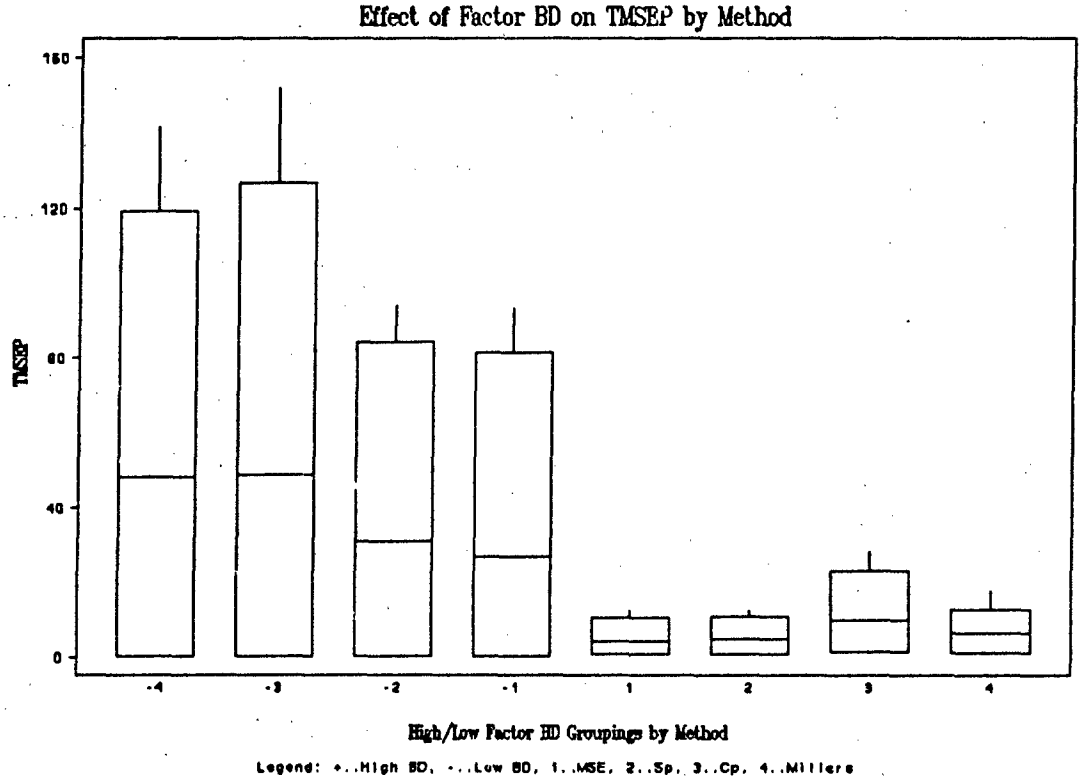


Figure 15. Box and Whisker Plots Showing the Effect of Factor BD on TMSEP by Method

This research proposes a two-stage variable selection technique. Miller's method is used to first screen the variable pool and reduce the number of extraneous variables. Next the Minimum MSE method is used to select the model from this reduced variable pool.

Factor C, the variance of the extraneous variables, had little or no effect on either PM or TMSEP. It was the only factor which had no impact throughout this research effort.

Neither did it appear in any of the PM or TMSEP equations. Based on these results, this factor could be dropped from further consideration.

Another useful result of this research is the comparison of the two MSEP criteria: Minimum S_p and Minimum C_p. A great deal of praise has been given to the Minimum S_p criterion in the past 15 years. It was identified as one of the most promising methods when the regressors are random and one desires to minimize the mean square error of prediction. The minimum C_p criterion has also received praise for minimizing mean square error of prediction, but its usefulness is limited to cases where the regressors are fixed. Some have recommended that the Minimum C_p criterion not be used in practice.

The results of this thesis indicate that the Minimum S_p method outperformed the Minimum C_p method at every factor level, using both PM and TMSEP. No evidence was found to refute the assertion that the Minimum C_p criterion should not be used in practice. In fact, this research effort supports using Minimum S_p method instead of the Minimum C_p method, thereby improving the selection process.

Most other simulations have dealt with the number of correct variables chosen of those available. No provisions were made for circumstances in which a significant regressor is not included in the variable pool. Therefore, techniques

praised as good variable selection techniques may not be as appealing as originally thought. This appears to be the case with Minimum C_p. It should be noted, however, that Mallows C_p method (C_p-close-to-p) is not the same as the Minimum C_p method (C_p-close-to-zero). This Mallows C_p method, as originally proposed, was not studied in this thesis.

Recommendations for Further Research

This research effort lends itself to several follow-on studies. The methodology established by Hansen and the computer programming groundwork in this research project make embellishments and the use of more complex model a feasible task.

One area which leads to further research deals with expanding the number of factors under consideration. This research effort studied six factors, but many more could be added. The response surface region could be expanded to include negative correlation, larger sample sizes, and the spread of the variance on the independent or correct variables. The factors studied could also include an indicator variable to keep track of the effect of dropping a significant variable. That is, by including a variable to keep track of the difference between the full model and a model where a variable is dropped, one could quantify the effects of failing to collect data on all the significant variables. This research only collected information on the effects of

dropping a variable and it was assumed that if all the variables were present the techniques studied would perform better. However, to gain a better understanding of Miller's method, it would be worthwhile to quantify the effects of not including all significant variables in the variable pool. To implement this, factor C (the variance of the extraneous variables) which had no effect, could be replaced with the indicator variable described above. Thus, the information desired could be gained without increasing the size of the experimental design.

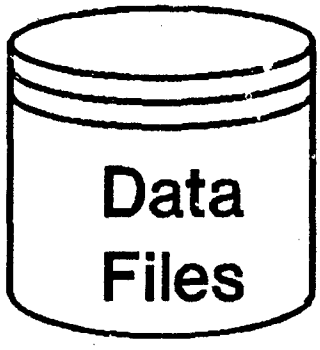
Further research could also be done to address the question of which screening and final selection method combinations work best together and under what circumstances. The four methods studied in this thesis could generate 16 screening and final selection method combinations. Some of these combinations may be eliminated a priori, but the rest could be studied either under the original six factors used in this thesis or under an expanded set of factors. The number of methods considered could also be increased. One method which could be added is Mallows C_p, as the method was originally set forth. This would allow a comparison between Miller's method and other variable selection techniques not studied in this thesis.

This thesis effort has implemented a promising new variable selection technique: Miller's method. Additionally, by comparing its performance with three well tested

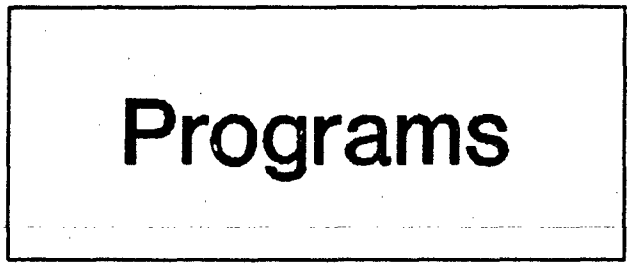
methods, this research has served to suggest a possible role for Miller's method among the many selection techniques. The results of this research indicate that Miller's method may be most effective when used as a screening method prior to final variable selection.

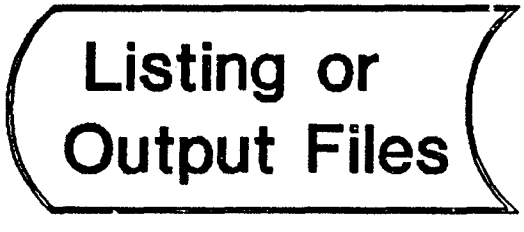
Appendix A: Flow Chart for the Development of the MSE, S_p, and C_p PMs

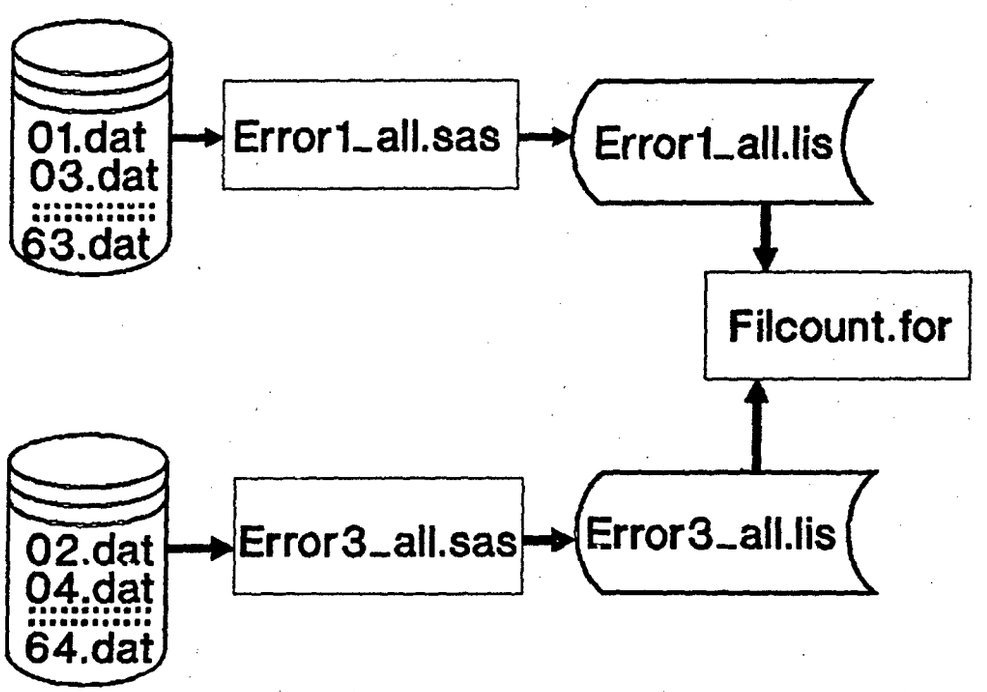
Legend of Flow Chart Symbols

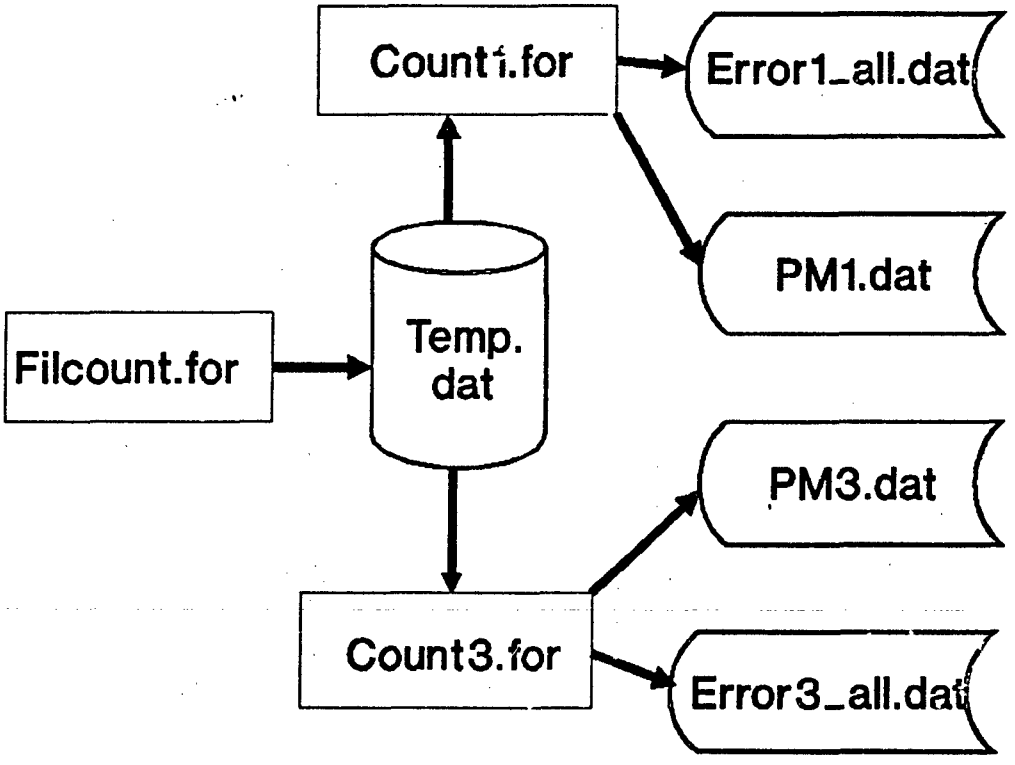


Flow of Data

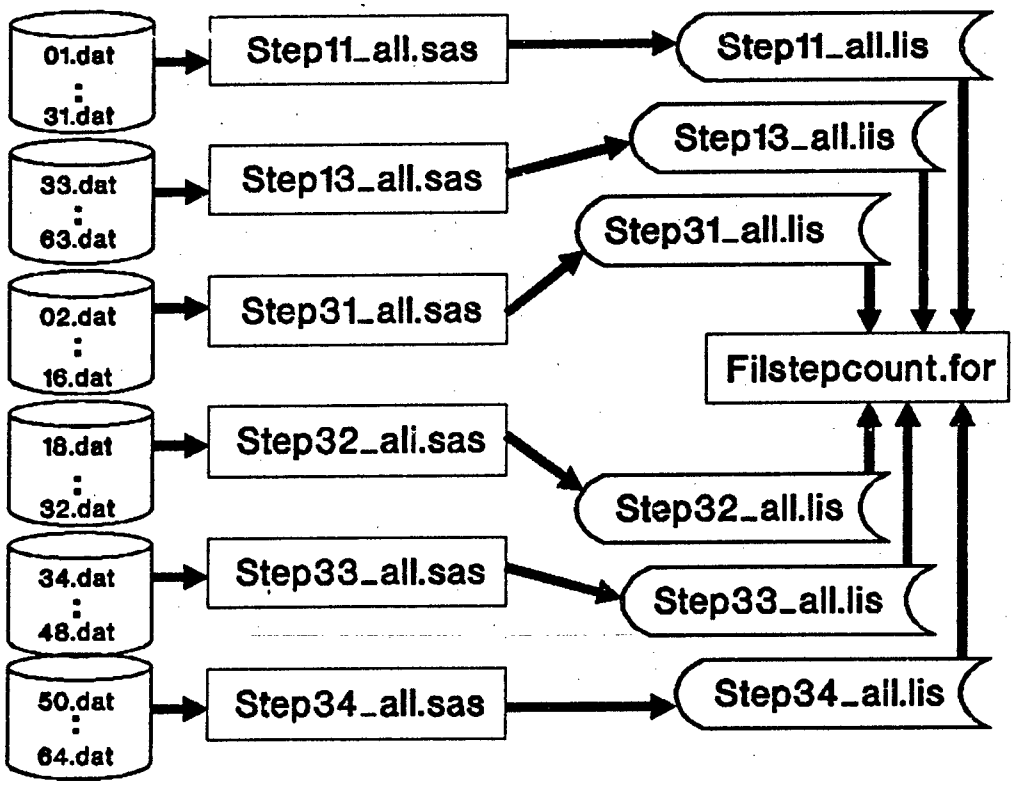


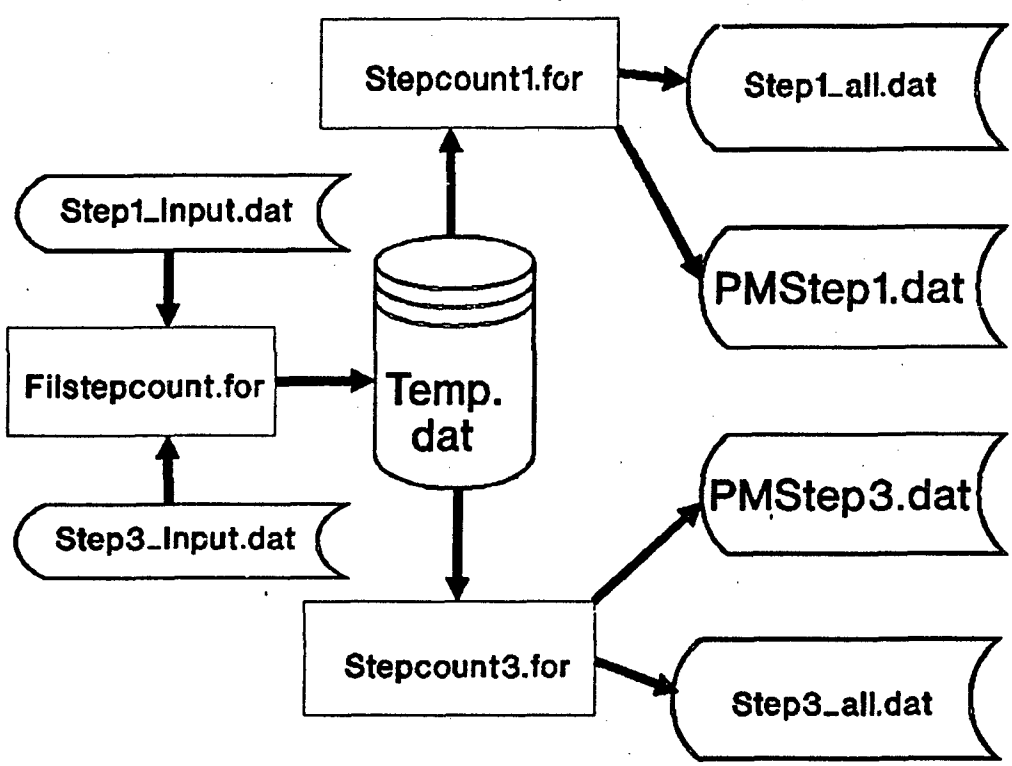



01.dat
03.dat
63.dat
Error1_all.sas
Error1_all.lis
Filcount.for
02.dat
04.dat
64.dat
Error3_all.sas
Error3_all.lis

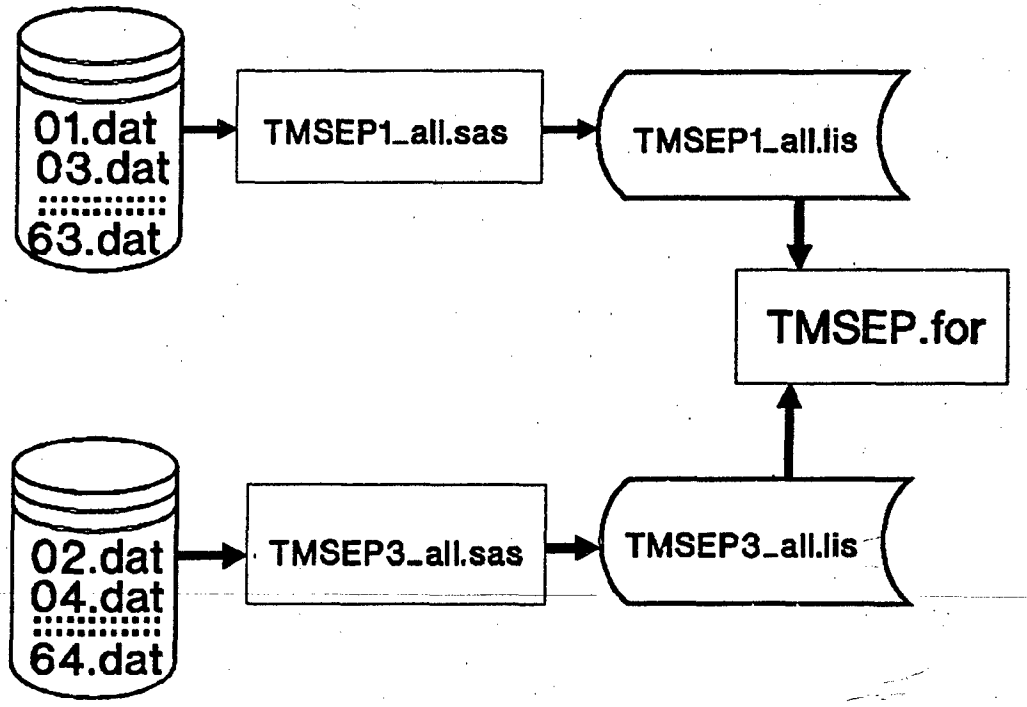

Count1.for
Error1_all.dat
PM1.dat
Filcount.for
Temp.
dat
PM3.dat
Count3.for
Error3_all.dat

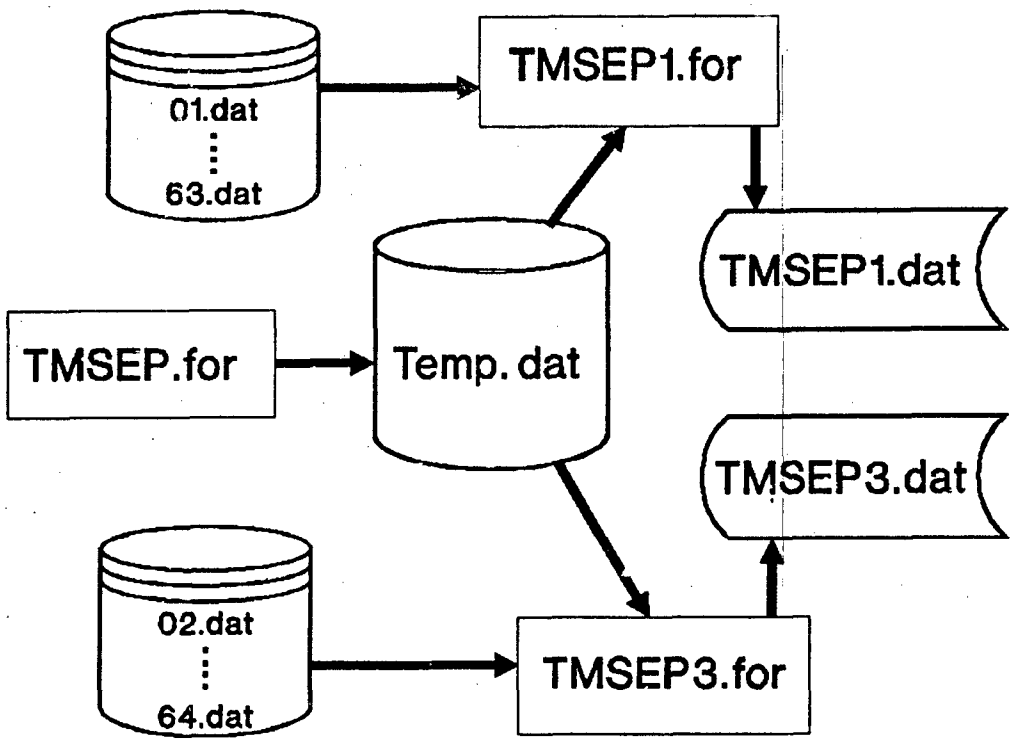
Appendix B: Flow Chart for the Development of the PM for Miller's Method



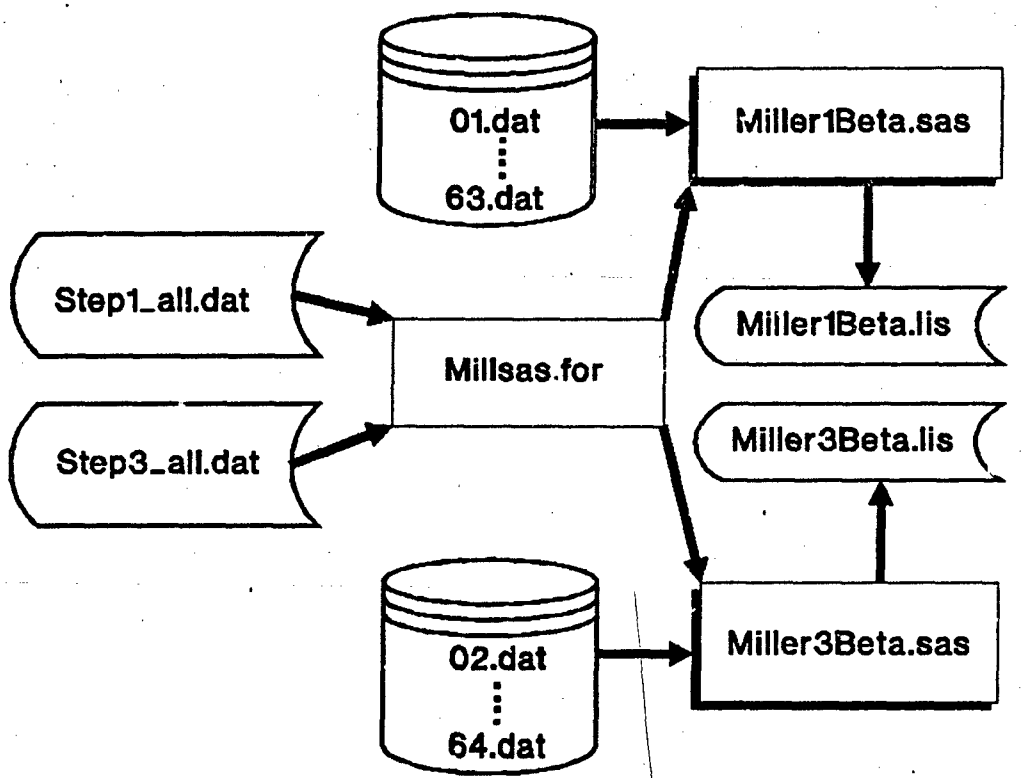

Stepcount1.for
Step1_all.dat
Step1_Input.dat
PMStep1.dat
Filstepcount.for
Temp.
dat
PMStep3.dat
Step3_Input.dat
Stepcount3.for
Step3_all.dat

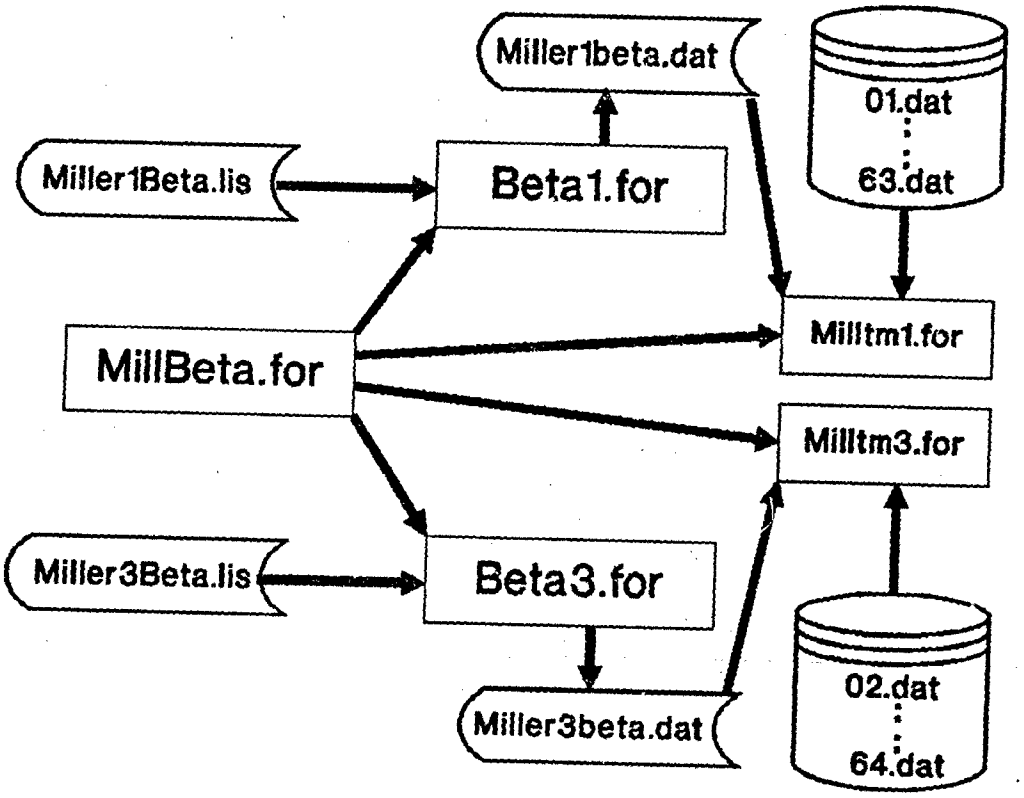
Appendix C: Flow Chart for the Development of the MSE, S_p, and C_p TMSEPs

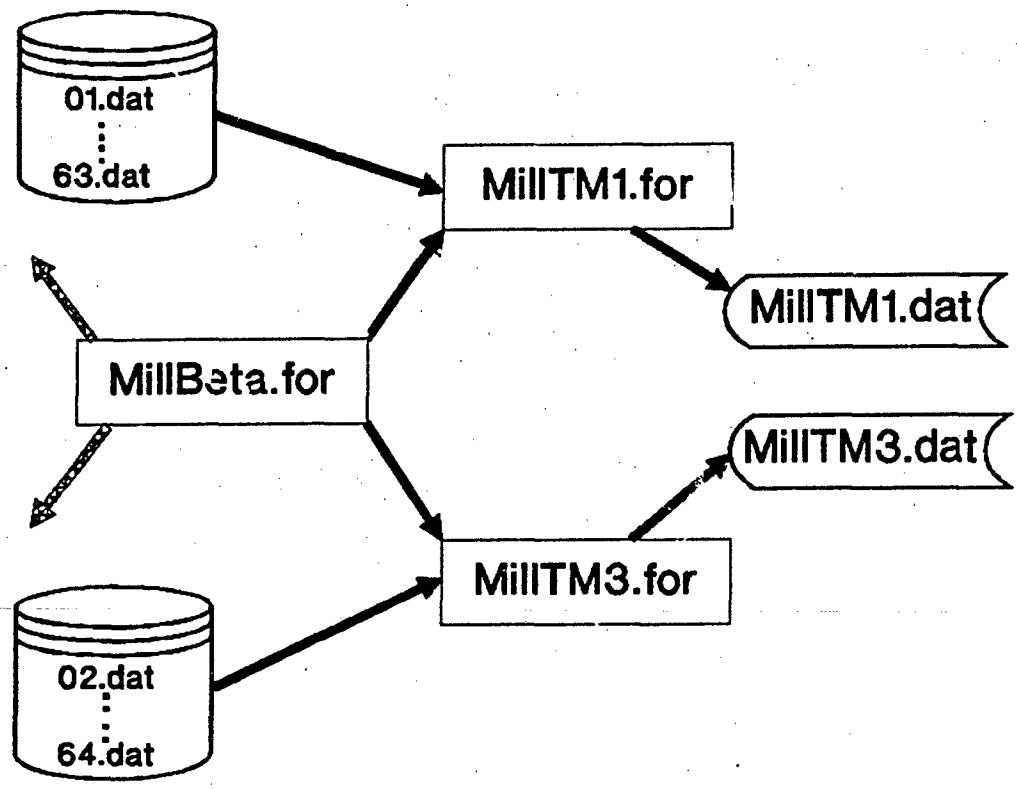



01.dat
63.dat
TMSEP1.for
TMSEP1.dat
TMSEP.for
Temp. dat
TMSEP3.dat
02.dat
64.dat
TMSEP3.for

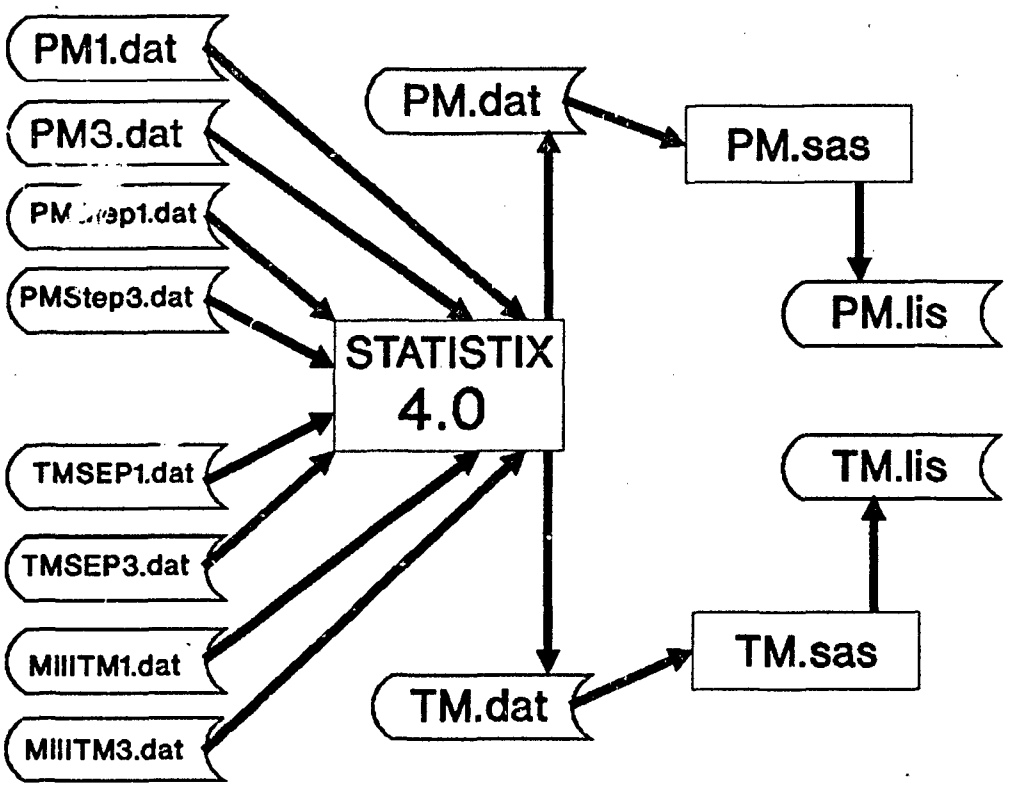
Appendix D: Flow Chart showing the Development of TMSEP for Miller's Method




Miller1beta.dat
01.dat
63.dat
Miller1Beta.lis
Beta1.for
MillBeta.for
Milltm1.for
Milltm3.for
Miller3Beta.lis
Beta3.for
02.dat
64.dat
Miller3beta.dat


01.dat
63.dat
MillTM1.for
MillTM1.dat
MillBeta.for
MillTM3.dat
MillTM3.for
02.dat
64.dat

Appendix E: Flowchart Describing the Experiments Using PM and TMSEP



Appendix F: A Glossary of Input/Output Data Files

Used Throughout All Sections

01.dat, 02.dat,...,64.dat
64 specifically generated data files, one file for each of the 64 permutations of the six-factors analysis

01.dat, 03.dat,...,63.dat
Of the 64 files, these are the ones with one extraneous variable

02.dat, 04.dat,...,64.dat
Of the 64 files, these are the ones with three extraneous variables

Temp.dat
Scratch file used to pass large amounts of data between FORTRAN main routines and their associated subroutines. Always contains temporary data generated by the most recently executed FORTRAN program.

Calculating PM for MSE, SP and CP methods

Error1_all.lis
Listing generated by SAS program Error1_all.sas. Contains output from the procedure RSquare (options MSE, SP, CP) run on 1920 data sets with one extraneous variable.

Error3_all.lis
Listing generated by SAS program Error1_all.sas. Contains output from the procedure RSquare (options MSE, SP, CP) run on 1920 data sets with three extraneous variables.

Error1_all.dat
Output from the FORTRAN subroutine Count1.for. Contains the selected model according to the MSE, SP, and CP methods for each of the 1920 data sets with one extraneous variable.

Error3_all.dat
Output from the FORTRAN subroutine Count3.for. Contains the selected model according to the MSE, SP, and CP methods for each of the 1920 data sets with three extraneous variables.

PM1.dat
Output from the FORTRAN subroutine Count1.for. Contains performance measures at each of the 32 odd design points

(1,3,...,63) for the MSE, SP, and CP methods of variable selection.

PM3.dat
Output from the FORTRAN subroutine Count3.for. Contains performance measures at each of the 32 even design points (2,4,...,64) for the MSE, SP, and CP methods of variable selection.

Calculating PM for Miller's method

Step11_all.lis
Listing generated by SAS program Step11_all.sas. Contains output from the procedure Stepwise(Forward Selection) run on 960 data sets (1,3,...,31) with one extraneous variable.

Step13_all.lis
Listing generated by SAS program Step11_all.sas. Contains output from the procedure Stepwise(Forward Selection) run on 960 data sets (33,35,...,63) with one extraneous variable.

Step31_all.lis
Listing generated by SAS program Step31_all.sas. Contains output from the procedure Stepwise(Forward Selection) run on 480 data sets (2,4,...,16) with three extraneous variables.

Step32_all.lis
Listing generated by SAS program Step32_all.sas. Contains output from the procedure Stepwise(Forward Selection) run on 480 data sets (18,20,...,32) with three extraneous variables.

Step33_all.lis
Listing generated by SAS program Step33_all.sas. Contains output from the procedure Stepwise(Forward Selection) run on 480 data sets (34,36,...,48) with three extraneous variables.

Step34_all.lis
Listing generated by SAS program Step34_all.sas. Contains output from the procedure Stepwise(Forward Selection) run on 480 data sets (50,52,...,64) with three extraneous variables.

Step1_Input.dat
Input data file for FilStepCount.for. Contains the names of the SAS listing files (from data sets with one extraneous variable) that FilCount.for is to process.

Step3_Input.dat
Input data file for FilStepCount.for. Contains the names of the SAS listing files (from data sets with three extraneous variables) that FilCount.for is to process.

Step1_all.dat
Generated by FORTRAN subroutine StepCount1.for. Contains the model selected via Miller's method for each of 1920 data sets with one extraneous variable.

Step3_all.dat
Generated by FORTRAN subroutine StepCount3.for. Contains the model selected via Miller's method f each of 1920 data sets with three extraneous variables.

PMStep1.dat
Output from the FORTRAN subroutine StepCount1.for. Contains performance measures at each of the 32 odd design points (1,3,...,63) for the Miller's method of variable selection.

PMStep3.dat
Output from the FORTRAN subroutine StepCount3.for. Contains performance measures at each of the 32 even design points (2,4,...,64) for the Miller's method of variable selection.

Stepwise Analysis using PM for each method

PM.dat
Output from the statistical analysis-program STATISTIX 4.0. Contains the design point four columns of PMs (one for each method) augmented with a full 2^6 factorial design matrix. This file is then used as input to the SAS program PM.sas.

PM.lis
Listing file generated by the SAS program PM.sas. Contains the complete analysis from the procedure Stepwise. Attempts a best fit for each method's PM as a linear function of the six factors studied and their interactions.

Calculating TMSEP for MSE, SP and CP methods

TMSEP1_all.lis
Listing generated by SAS program TMSEP1_all.sas. Contains output from the procedure RSquare (options MSE, SP, CP, and B) run on 1920 data sets with one extraneous variable.

TMSEP3_all.lis

Listing generated by SAS program TMSEP3_all.sas. Contains output from the procedure RSquare (options MSE, SP, CP, and B) run on 1920 data sets with one extraneous variable.

TMSEP1.dat
Generated by FORTRAN subroutine TMSEP1.for. Contains the TMSEPs for the 32 odd design points (1,3,...,63) with one extraneous variable.

TMSEP3.dat
Generated by FORTRAN subroutine TMSEP3.for. Contains the TMSEPs for the 32 even design points (2,4,...,64) with three extraneous variables.

Calculating TMSEP for Miller's method

Miller1Beta.sas
Generated by FORTRAN program Mill1SAS.for. This is a SAS input program design to calculate the constant and the coefficients of regression for each of the 1920 models selected using Miller's method and data sets with one extraneous variable.

Miller3Beta.sas
Generated by FORTRAN program Mill3SAS.for. This is a SAS input program design to calculate the constant and the coefficients of regression for each of the 1920 models selected using Miller's method and data sets with three extraneous variables.

Miller1Beta.lis
Listing file generated by the SAS program Miller1Beta.sas. Contains the unformatted and unfiltered data on the constant and the coefficients of regression for each of the 1920 models selected using Miller's method and data sets with one extraneous variable.

Miller3Beta.lis
Listing file generated by the SAS program Miller3Beta.sas. Contains the unformatted and unfiltered data on the constant and the coefficients of regression for each of the 1920 models selected using Miller's method and data sets with three extraneous variables.

Miller1Beta.dat
Generated by the FORTRAN subroutine Beta1.sas. Contains the filtered and formatted data on the constant and the coefficients of regression for each of the 1920 models selected using Miller's method and data sets with one extraneous variable.

Miller3Beta.dat
Generated by the FORTRAN subroutine Beta3.sas. Contains the filtered and formatted data on the constant and the coefficients of regression for each of the 1920 models selected using Miller's method and data sets with three extraneous variables.

MillTM1.dat
Generated by the FORTRAN subroutine MillTM1.for. Contains the TMSEPs for the odd design points (1,3,...,63) with one extraneous variable.

MillTM3.dat
Generated by the FORTRAN subroutine MillTM3.for. Contains the TMSEPs for the odd design points (2,4,...,64) with three extraneous variables.

Stepwise Analysis using TMSEP for each method

TM.dat
Output from the statistical analysis program STATISTIX 4.0. Contains the design point four columns of TMSEPs (one for each method) augmented with a full 2^6 factorial design matrix. This file is then used as input to the SAS program TM.sas.

TM.lis
Listing file generated by the SAS program TM.sas. Contains the complete analysis from the procedure Stepwise. Attempts a best fit for each method's TMSEP as a linear function of the six factors studied and their interactions.

Appendix C: A Glossary of FORTRAN Program Files

Calculating PM for MSE, SP and CP methods

FilCount.for

PURPOSE: Filters the SAS RSquare listings and generates the formatted file Temp.dat of all the possible model combinations for each of the 3840 data set. Calls Count1 and Count3 to select the best model.

INPUT DATA FILES: Error1_all.lis, Error3_all.lis

OUTPUT DATA FILES: Temp.dat

SUBROUTINES CALLED: Count1.for, Count3.for

Count1.for

PURPOSE: Selects the best model for each of the 1920 data sets (one extraneous variable) from a file of all possible model combinations for each set. Uses the MSE, SP, and CP methods of variables selection. Calculates a performance measure for each of the three groups of 60 models selected at each of the odd design points (1,3,...,63).

INPUT DATA FILES: Temp.dat

OUTPUT DATA FILES: Error1_all.dat, PM1.dat

SUBROUTINES CALLED: None

Count3.for

PURPOSE: Selects the best model for each of the 1920 data sets (three extraneous variables) from a file of all possible model combinations for each set. Uses the MSE, SP, and CP methods of variables selection. Calculates a performance measure for each of the three groups of 60 models selected at each of the even design points (2,4,...,64).

INPUT DATA FILES: Temp.dat

OUTPUT DATA FILES: Error3_all.dat, PM3.dat

SUBROUTINES CALLED: None

Calculating PM for Miller's method

FilStepCount.for

PURPOSE: Filters the SAS Stepwise(Forward Selection) listings and generates the formatted file Temp.dat of the model selected for each of the 3840 data sets. Calls Stepcount1 and Stepcount3 to select the best models.

INPUT DATA FILES: Step11_all.lis, Step13_all.lis, Step31_all.lis, Step32_all.lis, Step33_all.lis, Step34_all.lis Step1_Input.dat, Step3_Input.dat

OUTPUT DATA FILES: Temp.dat

SUBROUTINES CALLED: Stepcount1.for, Stepcount3.for

StepCount1.for

PURPOSE: Implements Miller's method for each of the 1920 models (from data sets with one extraneous variable). Calculates a performance measure for each group of 60 models selected at each odd design point (1,3,...,63).

INPUT DATA FILES: Temp.dat

OUTPUT DATA FILES: Step1_all.dat, PMStep1.dat

SUBROUTINES CALLED: None

StepCount3.for

PURPOSE: Implements Miller's method for each of the 1920 models (from data sets with three extraneous variables). Calculates a performance measure for each group of 60 models selected at each even design point (2,4,...,64).

INPUT DATA FILES: Temp.dat

OUTPUT DATA FILES: Step3_all.dat, PMStep3.dat

SUBROUTINES CALLED: None

Calculating P for MSE, SP and CP methods

TMSEP.for

PURPOSE: Filters the SAS RSquare listings and generates the formatted file Temp.dat of all the possible model combinations for each of the 3840 data set. Calls TMSEP1 and TMSEP3 to select the best model.

INPUT DATA FILES: TMSEP1_all.lis, TMSEP3_all.lis

OUTPUT DATA FILES: Temp.dat

SUBROUTINES CALLED: TMSEP1.dat, TMSEP3.dat

TMSEP1.for

PURPOSE: Selects the best model for each of the 1920 data sets (one extraneous variable) from a file of all possible model combinations for each set. Uses the MSE, SP, and CP methods of variables selection. Using each of the data sets with one extraneous variable, it calculates a TMSEP for each of the three groups of 60 models selected at each of the odd design points (1,3,...,63).

INPUT DATA FILES: 01.dat, 03.dat,...,63.dat, Temp.dat

OUTPUT DATA FILES: TMSEP1.dat

SUBROUTINES CALLED: None

TMSEP3.for

PURPOSE: Selects the best model for each of the 1920 data sets (three extraneous variables) from a file of all possible model combinations for each set. Uses the MSE, SP, and CP methods of variables selection. Using each of the data sets with three extraneous variables, it calculates a TMSEP for each of the three groups of 60 models selected at each of the even design points (2,4,...,64).

INPUT DATA FILES: 02.dat, 04.dat,...,64.dat, Temp.dat

OUTPUT DATA FILES: TMSEP3.dat

SUBROUTINES CALLED: None

Calculating TMSEP for Miller's method

MillSAS.for

PURPOSE: Reads the 3840 models (selected by Miller's method) and generates SAS code, specific to each model, to estimate the constant term and the coefficients of regression for that model.

INPUT DATA FILES: Step1_all.dat, Step3_all.dat

OUTPUT DATA FILES: Miller1Beta.sas, Miller3Beta.sas

SUBROUTINES CALLED: None

MillBeta.for

PURPOSE: Calls Beta1 and Beta3 and then calls MillTM1 and MillTM3.

INPUT DATA FILES: None

OUTPUT DATA FILES: None

SUBROUTINES CALLED: Beta1.for, Beta3.for, MillTM1.for, MillTM3.for

Beta1.for

PURPOSE: Filters the unformatted SAS listing file produced by the SAS program Miller1Beta.sas (from data sets with one extraneous variable) and outputs the estimates of the model constant and regression coefficients in a sorted, formatted order.

INPUT DATA FILES: Miller1Beta.lis

OUTPUT DATA FILES: Miller1Beta.dat

SUBROUTINES CALLED: None

Beta3.for

PURPOSE: Filters the unformatted SAS listing file produced by the SAS program Miller3Beta.sas (from data sets with three extraneous variables) and outputs the estimates of the

model constant and regression coefficients in a sorted, formatted order.

INPUT DATA FILES: Miller3Beta.lis

OUTPUT DATA FILES: Miller3Beta.dat

SUBROUTINES CALLED: None

Mill1TM1.for

PURPOSE: At each of the 32 odd design points (1,3,...,63, the ones with only one extraneous variable) it examines each of the 60 model predicted and calculates a aggregated TMSEP for that design point.

INPUT DATA FILES: Miller1Beta.dat, 01.dat,....,63.dat

OUTPUT DATA FILES: Mill1TM1.dat

SUBROUTINES CALLED: None

Mill1TM3.for

PURPOSE: At each of the 32 even design points (2,4,...,64, the ones with only three extraneous variables) it examines each of the 60 model predicted and calculates a aggregated TMSEP for that design point.

INPUT DATA FILES: Miller3Beta.dat, 02.dat, 04.dat,...,64.dat

OUTPUT DATA FILES: Mill1TM3.dat

SUBROUTINES CALLED: None

BARR.FOR

PURPOSE: Written to read and correct most errors found in Hansen's data files. It was written by Dr. David Barr. It scans the data file after correction and outputs certain data characteristics for verification. Some errors had to be corrected by hand, but this program will allow the experimenter to be absolutely certain about the data's current characteristics.

Appendix H: A Glossary of SAS Program Files

Calculating PM for MSE, SP and CP methods

Error1_all.sas
Reads 01.dat,03.dat,...,63.dat by set and use the RSquared procedure to generate all-possible models for each set of 10 or 20. MSE, SP, and CP statistics are calculated for each model. The listing file Error1_all.lis is output.

Error3_all.sas
Reads 02.dat,04.dat,...,64.dat by set and use the RSquared procedure to generate all-possible models for each set of 10 or 20. MSE, SP, and CP statistics are calculated for each model. The listing file Error3_all.lis is output.

Calculating PM for Miller's method

Step11_all.sas
Reads 01.dat,03.dat,...,31.dat by set and augments each set with four known random predictors. The Stepwise procedure is than run and one model is chosen for each set. The listing Step11_all.lis is generated.

Step13_all.sas
Reads 33.dat,35.dat,...,63.dat by set and augments each set with four known random predictors. The Stepwise procedure is than run and one model is chosen for each set. The listing Step13_all.lis is generated.

Step31_all.sas
Reads 02.dat,04.dat,...,16.dat by set and augments each set with six known random predictors. The Stepwise procedure is than run and one model is chosen for each set. The listing Step31_all.lis is generated.

Step32_all.sas
Reads 18.dat,20.dat,...,32.dat by set and augments each set with six known random predictors. The Stepwise procedure is than run and one model is chosen for each set. The listing Step32_all.lis is generated.

Step33_all.sas
Reads 34.dat,36.dat,...,48.dat by set and augments each set with six known random predictors. The Stepwise procedure is than run and one model is chosen for each set. The listing Step33_all.lis is generated.

Step34_all.sas
Reads 50.dat,52.dat,...,64.dat by set and augments each set with six known random predictors. The Stepwise procedure is than run and one model is chosen for each set. The listing Step34_all.lis is generated.

Stepwise analysis using PM for each method

PM.sas
Reads PM.dat and performs four separate Stepwise regressions. Each regression considers a different dependent variable but the uses the same independent variables. Generates listing file PM.lis.

Calculating TMSEP for MSE, SP and CP methods

TMSEP1_all.sas
Reads 01.dat,03.dat,...,63.dat by set and use the RSquared procedure to generate all-possible models for each set of 10 or 20. MSE, SP, and CP statistics and the coefficients of regression are calculated for each model. The listing file TMSEP1_all.lis is output.

TMSEP3_all.sas
Reads 02.dat,04.dat,...,64.dat by set and use the RSquared procedure to generate all-possible models for each set of 10 or 20. MSE, SP, and CP statistics and the coefficients of regression are calculated for each model. The listing file TMSEP3_all.lis is output.

Calculating TMSEP for Miller's method

Miller1Beta.sas
Reads 01.dat, 03.dat,...,63.dat by set and uses the RSquared procedure (with various switches) to calculate the coefficients of regression for only the model selected for each data set by Miller's method. The listing file Miller1Beta-.lis is generated.

Miller3Beta.sas
Reads 02.dat, 04.dat,...,64.dat by set and uses the RSquared procedure (with various switches) to calculate the coefficients of regression for only the model selected for each data set by Miller's method. The listing file Miller3Beta-.lis is generated.

Stepwise analysis using TMSEP for each method

TM.sas
Reads TM.dat and performs four separate Stepwise regressions. Each regression considers a different dependent variable but the uses the same independent variables. Generates listing file TM.lis.

Appendix I: FORTRAN Programs

List of FORTRAN Programs

```
************************************************************
*                          BARR.FOR
*
* This program reads in Hansen's data files and scans them
* for certain data characteristics.  These are output for
* verification.
************************************************************

      real x(4),ex(3)
      integer n,set,count,ind(6),inum,nex,ss,nexref(2)
      integer lecount,hecount,lxcount,hxcount
      double precision
     + errorsum,error2sum,lerrorsum,lerror2sum
      double precision exsum(3),ex2sum(3),xsum(4),x2sum(4)
      double precision herrorsum,herror2sum,lexsum,lex2sum
      double precision
     + hexsum,hex2sum,lxsum,lx2sum,hxsum,hx2sum
      double precision
     + xprod(4,4),lxsum,lx2sum,hxsum,hx2sum,
     + error
      character*6 nameoffile(64)

      nameoffile(1)='01.dat'
      nameoffile(2)='02.dat'
      nameoffile(3)='03.dat'
      nameoffile(4)='04.dat'
      nameoffile(5)='05.dat'
      nameoffile(6)='06.dat'
      nameoffile(7)='07.dat'
      nameoffile(8)='08.dat'
      nameoffile(9)='09.dat'
      nameoffile(10)='10.dat'
      nameoffile(11)='11.dat'
      nameoffile(12)='12.dat'
      nameoffile(13)='13.dat'
      nameoffile(14)='14.dat'
      nameoffile(15)='15.dat'
      nameoffile(16)='16.dat'
      nameoffile(17)='17.dat'
      nameoffile(18)='18.dat'
      nameoffile(19)='19.dat'
      nameoffile(20)='20.dat'
      nameoffile(21)='21.dat'
      nameoffile(22)='22.dat'
      nameoffile(23)='23.dat'
      nameoffile(24)='24.dat'
      nameoffile(25)='25.dat'
      nameoffile(26)='26.dat'
      nameoffile(27)='27.dat'
      nameoffile(28)='28.dat'
      nameoffile(29)='29.dat'
      nameoffile(30)='30.dat'
```

```
      nameoffile(31)='31.dat'
      nameoffile(32)='32.dat'
      nameoffile(33)='33.dat'
      nameoffile(34)='34.dat'
      nameoffile(35)='35.dat'
      nameoffile(36)='36.dat'
      nameoffile(37)='37.dat'
      nameoffile(38)='38.dat'
      nameoffile(39)='39.dat'
      nameoffile(40)='40.dat'
      nameoffile(41)='41.dat'
      nameoffile(42)='42.dat'
      nameoffile(43)='43.dat'
      nameoffile(44)='44.dat'
      nameoffile(45)='45.dat'
      nameoffile(46)='46.dat'
      nameoffile(47)='47.dat'
      nameoffile(48)='48.dat'
      nameoffile(49)='49.dat'
      nameoffile(50)='50.dat'
      nameoffile(51)='51.dat'
      nameoffile(52)='52.dat'
      nameoffile(53)='53.dat'
      nameoffile(54)='54.dat'
      nameoffile(55)='55.dat'
      nameoffile(56)='56.dat'
      nameoffile(57)='57.dat'
      nameoffile(58)='58.dat'
      nameoffile(59)='59.dat'
      nameoffile(60)='60.dat'
      nameoffile(61)='61.dat'
      nameoffile(62)='62.dat'
      nameoffile(63)='63.dat'
      nameoffile(64)='64.dat'
      nexref(1)=1
      nexref(2)=3

1000  format(5x,4(f15.5,1x))

      open(unit=8,file='iv1.out',status='new')
      open(unit=9,file='iv0.out',status='new')
      open(unit=11,file='dbarr.out',status='new')
      open(unit=12,file='dbarr.log',status='new')
      k=64
      ind(3)=0
      ind(4)=0
      ind(5)=0
      ind(6)=0
      count=0
      lxcount=0
      hxcount=0
      lecount=0
```

```
        hecount=0
        lxsum=0
        lx2sum=0
        hxsum=0
        hx2sum=0
        lerrorsum=0
        lerror2sum=0
        herrorsum=0
        herror2sum=0

*****************************************************************

        do 10 inum=1,k

        errorsum=0
        error2sum=0

        do 11 i11=1,3
        exsum(i11)=0
        ex2sum(i11)=0
11     continue

        do 12 i12=1,4
        xsum(i12)=0
        x2sum(i12)=0
12     continue

        do 13 i13=1,4
        do 14 i14=1,4
        xprod(i13,i14)=0
14     continue
13     continue

        print *, nameoffile(inum)
        write(11,*) nameoffile(inum)
        write(12,*) nameoffile(inum)
        ind(1)=inum+1-2*((inum+1)/2)
       ind(2)=iabs(2*((inum+3)/4)-((inum+1)/2)-1)
       ind(3)=iabs(2*((inum+7)/8)-((inum+3)/4)-1)
       ind(4)=iabs(2*((inum+15)/16)-((inum+7)/8)-1)
       ind(5)=iabs(2*((inum+31)/32)-((inum+15)/16)-1)
       ind(6)=iabs(2*((inum+63)/64)-((inum+31)/32)-1)
       nex=nexref(ind(1)+1)

        open(unit=10,file=nameoffile(inum),status='old')

        if (ind(5).eq.0) then
          ss=10
        else
       ss=20
        endif
```

```
        n=ss*60
        do 50 h= 1,n
          read (10,*) set,y, (x(i),i=1,4), (ex(i),i=1,nex)

       count=count+1
       if(ind(6).eq.0) then
         lecount=lecount+1
       else
         hecount=hecount+1
       endif

       if(nex.eq.1) then
         ex(2)=0
         ex(3)=0
         endif


        call errorcomp(x,y,error,errorsum,error2sum)
        call extra(nex,ex,exsum,ex2sum)
        call xi(x,xsum,x2sum,xprod)

          write(11,*) set,y,error,(x(i),i=1,4),(ex(i),i=1,3),
     + (ind(7-i),i=1,6)
50      continue

        call
     + endprint(n,ind,nex,ss,errorsum,error2sum,lerrorsum,
     + lerror2sum,exsum,ex2sum,xsum,x2sum,xprod,herrorsum,
     + herror2sum,
     + lxsum,lx2sum,hxsum,hx2sum,lxcount,
     + hxcount,nameoffile,inum)
10      continue

*****************************************************************

        print*, ' '
        write(12,*) ' '
        print *, 'number of observations = ',count
        write(12,*) 'number of observations = ',count

        print *, 'small independent variance = ',
     + lx2sum/(lxcount)-(lxsum/(lxcount))**2
        print *, 'large independent variance = ',
     + hx2sum/(lxcount)-(hxsum/(lxcount))**2
        write(12,*) 'small independent variance = ',
     + lx2sum/(lxcount)-(lxsum/(lxcount))**2
        write(12,*) 'large independent variance = ',
     + hx2sum/(hxcount)-(hxsum/(hxcount))**2
        print *, 'small error variance = ',
     + lerror2sum/lecount-(lerrorsum/lecount)**2
        write(12,*) 'small error variance = ',
     + lerror2sum/lecount-(lerrorsum/lecount)**2
```

```
        print *, 'large error variance = ',
     + herror2sum/hecount-(herrorsum/hecount)**2
        write(12,*) 'large error variance = ',
     + herror2sum/hecount-(herrorsum/hecount)**2

        END

*****************************************************************

      subroutine errorcomp(x,y,error,errorsum,error2sum)

       double precision error,errorsum,error2sum
       real x(4)

      yactual=0
         do 60 p=1,4
            yactual = yactual+x(p)
60       continue
      error=y-yactual
      errorsum=errorsum+error
      error2sum=error2sum+error*error

      return
       END
*****************************************************************

       subroutine endprint(n,ind,nex,ss,errorsum,error2sum,
     + lerrorsum,lerror2sum,exsum,ex2sum,xsum,x2sum,xprod,
     + herrorsum,herror2sum,
     + lxsum,lx2sum,hxsum,hx2sum,lxcount,
     + hxcount,nameoffile,inum)

       integer n,ind(6),nex,ss,lxcount,hxcount,inum
       double precision errorsum,error2sum,lerrorsum,
     + lerror2sum,exsum(3),ex2sum(3),xsum(4),x2sum(4),
     + xprod(4,4),r(4,4),v(4),herrorsum,herror2sum,ev(3),
     + lxsum,lx2sum,hxsum,hx2sum,rsum,ex1,ex2
       character*6 nameoffile(64)

1000   format(5x,4(f15.5,1x))
1010   format(5x,4(i10,1x))

      print *, (ind(7-i),i=1,6)
      write(12,*) (ind(7-i),i=1,6)
      print *, ' ',ind(1),' there are ',nex,' extraneous
    + variables'
      write(12,*) ' ',ind(1),' there are ',nex,' extraneous
    + variables'

      do 50 k=1,4
      v(k)=x2sum(k)/n-(xsum(k)/n)**2
50    continue
```

```
      rsum=0
      do 30 i=1,4
      do 40 j=1,i
      rn=xprod(i,j)/n-(xsum(i)/n)*(xsum(j)/n)
      r(i,j)=rn/sqrt(v(i)*v(j))
      r(j,i)=r(i,j)
      if(i.ne.j) then
       rsum=rsum+r(i,j)
      endif
40    continue
30    continue

      print *, ' ',ind(2),' correlation ',rsum/6
      write(12,*) ' ',ind(2),' correlation ',rsum/6

      do 35 i=1,4
      print *, (r(i,j),j=1,4)
      write(12,1000) (r(i,j),j=1,4)
35    continue


      ex1=0
      ex2=0
      do 10 j=1, nex
       ev(j)=ex2sum(j)/n-(exsum(j)/n)**2
       ex1=ex1+exsum(j)
       ex2=ex2+ex2sum(j)
10    continue
      ve=(ex2/(n*nex))-(ex1/(n*nex))**2
      print *, ' ',ind(3),' variance of extraneous ',ve
      write(12,*) ' ',ind(3),' variance of extraneous ',ve
      print *, (ev(k),k=1,nex)
      write(12,1000) (ev(k),k=1,nex)
      write(12,1000) (exsum(i),i=1,nex)
      write(12,1000) (ex2sum(i),i=1,nex)
      write(12,1010) n,n*nex

      x1=xsum(1)+xsum(2)+xsum(3)+xsum(4)
      x2=x2sum(1)+x2sum(2)+x2sum(3)+x2sum(4)
         vx=x2/(4*n)-(x1/(4*n))**2
      print *, ' ', ind(4),' variance of independent ',vx
       write(12,*) ' ', ind(4),' variance of independent ',vx

        if(vx.lt..00125) then
          lxsum=lxsum+x1
          lx2sum=lx2sum+x2
        lxcount=lxcount+4*n
        write(9,*) nameoffile(inum),ind(4),
     + x2/(4*n)-(x1/(4*n))**2
        else
```

```
          hxsum=hxsum+x1
          hx2sum=hx2sum+x2
        hxcount=hxcount+4*n
        write(8,*) nameoffile(inum),ind(4),
     + x2/(4*n)-(x1/(4*n))**2
        endif

        print *, ' ',ind(5),' the sample size is ',ss
        write(12,*) ' ',ind(5),' the sample size is ',ss

        print *, ' ',ind(6),' error variance =',
     + error2sum/n-(errorsum/n)**2
        write(12,*) ' ',ind(6),' error variance =',
     + error2sum/n-(errorsum/n)**2

        if(ind(6).eq.0) then
        lerrorsum=lerrorsum+errorsum
        lerror2sum=lerror2sum+error2sum
        else
        herrorsum=herrorsum+errorsum
        herror2sum=herror2sum+error2sum
        endif

        close (unit=10)
        return
        END
*****************************************************************

        subroutine extra(nex,ex,exsum,ex2sum)
        integer nex
        real ex(3)
        double precision exsum(3),ex2sum(3)
        do 10 i=1,nex
        exsum(i)=exsum(i)+ex(i)
        ex2sum(i)=ex2sum(i)+ex(i)*ex(i)
10     continue
        return

        END
*****************************************************************
        subroutine xi(x,xsum,x2sum,xprod)
        real x(4)
        double precision xsum(4),x2sum(4),xprod(4,4)
        do 10 i=1,4
        xsum(i)=xsum(i)+x(i)
        do 20 j=1,4
        xprod(i,j)=xprod(i,j)+x(i)*x(j)
20     continue
10     continue
        do 30 k=1,4
        x2sum(k)=xprod(k,k)
30     continue
```

```
      return

      END
*****************************************************************
```

```
*****************************************************************
****************** FORTRAN PROGRAM BETA1.FOR*********************
*****************************************************************

      SUBROUTINE BETA1

      INTEGER VARNUM,MODELNUM,J,K,L,N,P,TOTALLINES,CHARPOS
      REAL R2, B0, BETA(4), SORTED_BETAS(4)

      CHARACTER*132 LINE
      CHARACTER*2 MODEL(4)

      TOTALLINES=0

      OPEN (unit=10, file='miller1beta.lis',status='OLD',
     +                iostat=IERROR,err=1500)
      OPEN (unit=11, file='miller1beta.dat',status='NEW',
     +                iostat=IERROR,err=1500)

5     CONTINUE
      READ (10,900,END=888) LINE
900   FORMAT (A132)

      DO 10 J=1,132
          IF (LINE(J:J).EQ.'I') THEN
          CHARPOS = J
          GO TO 20
        ENDIF
10    CONTINUE
      GO TO 5

20    CONTINUE

        DO 35 L=1,4
          SORTED_BETAS(L)=0.0
          BETA(L)=0.0
35      CONTINUE


      IF ((LINE((CHARPOS+1):(CHARPOS+1))).EQ.'N') THEN
        READ (10,*)
        READ (10,*,END=1300) VARNUM, R2, B0
        VARNUM = VARNUM-1
        IF (VARNUM.GT.0) GO TO 1200
        WRITE (11,902) VARNUM, B0
902     FORMAT
     +(1X,I1,5X,F9.5,7X,'0.00000',7X,'0.00000',7X,'0.00000',
     +           7X,'0.00000')
        TOTALLINES=TOTALLINES+1

      ELSE
        IF ((LINE((CHARPOS+1):(CHARPOS+1))).EQ.'n') THEN
```

```
            K=1
              DO 30 J=CHARPOS,132
             IF(((LINE(J:J)).EQ.'X').OR.((LINE(J:J)).EQ.'E'))
            THEN
                  MODEL(K) = LINE(J:(J+1))
                  K = K+1
                ENDIF
30            CONTINUE
            MODELNUM = K-1
            IF (MODELNUM.LT.1) GO TO 1100
            READ (10,*)
            VARNUM=0
            READ (10,*,END=1300)VARNUM, R2,
     +           B0,(BETA(N),N=1,VARNUM)
            IF (VARNUM.NE.MODELNUM) GO TO 1000


            DO 40 P=1,VARNUM
              IF (MODEL(P).EQ.'X1') SORTED_BETAS(1)=BETA(P)
              IF (MODEL(P).EQ.'X2') SORTED_BETAS(2)=BETA(P)
              IF (MODEL(P).EQ.'X3') SORTED_BETAS(3)=BETA(P)
              IF (MODEL(P).EQ.'E1') SORTED_BETAS(4)=BETA(P)
40          CONTINUE

            WRITE (11,901) VARNUM, B0,(SORTED_BETAS(N),N=1,4)
901     FORMAT(1X,I1,5X,F9.5,5X,F9.5,5X,F9.5,5X,F9.5,5X,F9.5)
            TOTALLINES = TOTALLINES+1
          ENDIF
        ENDIF
        GO TO 5

888     CONTINUE
        CLOSE(10)
        CLOSE(11)
        PRINT *, 'FILTERING OF MILLER1BETA.LIS IS COMPLETE.'
        PRINT *, TOTALLINES,' LINES WRITTEN TO
      + MILLER1BETA.DAT.'
        PRINT *,' '
        GO TO 1600

1000    CONTINUE
        PRINT *,'Unexpected file format!',
     +          '  # of variable names does not',
     +          'correspond to # of varibles read.'
        GO TO 1600

1100    CONTINUE
        PRINT *,'Unexpected file format!',
     +          '  Could not find X1, X2, X3, or E1.'
        GO TO 1600

1200    CONTINUE
```

```
        PRINT *,'Unexpected file format!  Expecting ONLY B0.'

        GO TO 1600

1300    CONTINUE
        PRINT *,'Unexpected file format!  Encountered EOF
     + while ',
     +          'attempting to read VARNUM, R2, B0, and/or
     + Betas.'
        GO TO 1600

1500    CONTINUE
        PRINT 1501,'+++ ERROR WHILE OPENING FILE +++',
     +                '       error code = ',IERROR
1501    FORMAT (/1X, A/ 1X, A, I8/)

1600    CONTINUE
        END
```

```
**********************************************************************
***************** FORTRAN PROGRAM BETA3.FOR*****************
**********************************************************************


        SUBROUTINE BETA3

        INTEGER VARNUM,MODELNUM,J,K,L,N,P,TOTALLINES,CHARPOS
        REAL R2, B0, BETA(6), SORTED_BETAS(6)

        CHARACTER*132 LINE
        CHARACTER*2 MODEL(6)

        TOTALLINES=0

        OPEN (unit=12, file='miller3beta.lis',status='OLD',
     +                  iostat=IERROR,err=1500)
        OPEN (unit=13, file='miller3beta.dat',status='NEW',
     +                  iostat=IERROR,err=1500)

5       CONTINUE
        READ (12,900,END=888) LINE
900     FORMAT (A132)

        DO 10 J=1,132
            IF (LINE(J:J).EQ.'I') THEN
            CHARPOS = J
            GO TO 20
          ENDIF
10      CONTINUE
        GO TO 5

20      CONTINUE

          DO 35 L=1,6
            SORTED_BETAS(L)=0.0
            BETA(L)=0.0
35        CONTINUE


        IF ((LINE((CHARPOS+1):(CHARPOS+1))).EQ.'N') THEN
          READ (12,*)
          READ (12,*,END=1300) VARNUM, R2, B0
          VARNUM = VARNUM-1
          IF (VARNUM.GT.0) GO TO 1200
          WRITE (13,902) VARNUM, B0
902       FORMAT (1X,I1,5X,F9.5,7X,'0.00000',7X,'0.00000',
     +            7X,'0.00000',7X,'0.00000',7X,'0.00000',
     +            7X,'0.00000')
          TOTALLINES=TOTALLINES+1

        ELSE
```

```
          IF ((LINE((CHARPOS+1):(CHARPOS+1))).EQ.'n') THEN
            K=1
              DO 30 J=CHARPOS,132.
         IF (((LINE(J:J)).EQ.'X').OR.((LINE(J:J)).EQ.'E'))
         THEN
                    MODEL(K) = LINE(J:(J+1))
                    K = K+1
                  ENDIF
30              CONTINUE
               MODELNUM = K-1
               IF (MODELNUM.LT.1) GO TO 1100
               READ (12,*)
               VARNUM=0
               READ (12,*,END=1300)VARNUM, R2,
     +              B0,(BETA(N),N=1,VARNUM)
               IF (VARNUM.NE.MODELNUM) GO TO 1000


               DO 40 P=1,VARNUM
                 IF (MODEL(P).EQ.'X1') SORTED_BETAS(1)=BETA(P)
                 IF (MODEL(P).EQ.'X2') SORTED_BETAS(2)=BETA(P)
                 IF (MODEL(P).EQ.'X3') SORTED_BETAS(3)=BETA(P)
                 IF (MODEL(P).EQ.'E1') SORTED_BETAS(4)=BETA(P)
                 IF (MODEL(P).EQ.'E2') SORTED_BETAS(5)=BETA(P)
                 IF (MODEL(P).EQ.'E3') SORTED_BETAS(6)=BETA(P)
40             CONTINUE

               WRITE (13,901) VARNUM, B0,
     + (SORTED_BETAS(N),N=1,6)
901      FORMAT (1X,I1,5X,F9.5,5X,F9.5,5X,F9.5,5X,F9.5,5X,
     +              F9.5,5X,F9.5,5X,F9.5)
             TOTALLINES = TOTALLINES+1
           ENDIF
         ENDIF
         GO TO 5

888      CONTINUE
         CLOSE(12)
         CLOSE(13)
         PRINT *, 'FILTERING OF MILLER3BETA.LIS IS COMPLETE.'
         PRINT *, TOTALLINES,' LINES WRITTEN TO
     + MILLER3BETA.DAT.'
         PRINT *,' '
         GO TO 1600

1000     CONTINUE
         PRINT *,'Unexpected file format!',
     +          '  # of variable names does not',
     +         'correspond to # of varibles read.'
         GO TO 1600

1100     CONTINUE
```

```
      PRINT *,'Unexpected file format!',
     +        '  Could not find X1, X2, X3, E1, E2, E3.'

      GO TO 1600

1200  CONTINUE
      PRINT *,'Unexpected file format!  Expecting ONLY B0.'

      GO TO 1600

1300  CONTINUE
      PRINT *,'Unexpected file format!  Encountered EOF
     + while ',
     +        'attempting to read VARNUM, R2, B0, and/or
     + Betas.'
      GO TO 1600

1500  CONTINUE
      PRINT 1501,'+++ ERROR WHILE OPENING FILE +++',
     +           '      error code = ',IERROR
1501  FORMAT (/1X, A/ 1X, A, I8/)

1600  CONTINUE
      END
```

```
*****************************************************************
********************** FORTRAN PROGRAM COUNT1.FOR**********
*****************************************************************

      SUBROUTINE Count1 (NewOut)

        integer num(15),i,j,k,ptrmse,ptrsp
        integer ptrcp,varsmse,varssp,varscp
        integer check(4)
        integer n,emse,esp,ecp,DesignPoint
        integer ccp,cmse,csp,cumemse,cumecp,cumesp
        integer chartmse(0:3,0:3),chartcp(0:3,0:3)
        integer chartsp(0:3,0:3)
        real MSE(15),Sp(15),cp(15),r2(15)
        real minmse,minsp,mincp,nummse,numcp,numsp
        real mseeer,cpeer,speer
        real msepm,cppm,sppm
        character*2 m(4,15)
        character*20 NewOut
        check(1)=1
        check(2)=5
        check(3)=11
        check(4)=15
        DesignPoint=1

           do 7 i = 0,3
             do 3 k = 0,3
               chartmse(i,k)=0
               chartcp(i,k)=0
               chartsp(i,k)=0
3             continue
7             continue

        varsmse=0
        varssp =0
        varscp =0
        cumemse=0
        cumesp=0
        cumecp=0

        open (unit=11, file='temp.dat', status='old',
     +       iostat=IERROR, err=1000)
        open (unit=12, file=NewOut, status='new',
     +        iostat=IERROR, err=1000)
        open (unit=13, file='PM1.dat', status='new',
     +        iostat=IERROR, err=1000)
        write(13,*)'     DESIGNPOINT  MSE                ',
     +             'SP                    CP'


        Do 50 jj=1,63,2
           Write(12,*)' '
```

```
      Write(12,*)' '
      Write(12,*)' '
      Write(12,*) '******************** DESIGN POINT ',
     +              DesignPoint,' ********************'
      Write (12,*)' '
      DesignPoint=DesignPoint + 2

   do 20 k=1,60
   do 10 i=1,15
         emse=0
         esp=0
         ecp=0
         read(11,'(1X,I1)',end=40)num(i)

         IF (num(i).EQ.1) THEN
     +   read(11,905,end=40)num(i),r2(i),cp(i),MSE(i),Sp(i)
     +          ,m(1,i)
905      format(7X,I1,4X,F10.8,3X,F9.5,3X,F9.7,2X,
     +               F10.8,2X,A2)
         ELSE
         IF (num(i).EQ.2) THEN
     +   read(11,910,end=40)num(i),r2(i),cp(i),MSE(i),Sp(i)
     +          ,(m(j,i),j=1,2)
910      format(7X,I1,4X,F10.8,3X,F9.5,3X,F9.7,2X,
     +               F10.8,1X,2(1X,A2))
         ELSE

         IF (num(i).EQ.3) THEN
     +   read(11,915,end=40)num(i),r2(i),cp(i),MSE(i),Sp(i)
     +          ,(m(j,i),j=1,3)
915      format(7X,I1,4X,F10.8,3X,F9.5,3X,F9.7,2X,
     +               F10.8,1X,3(1X,A2))
         ELSE

         IF (num(i).EQ.4) THEN
     +   read(11,920,end=40)num(i),r2(i),cp(i),MSE(i),Sp(i)
     +          ,(m(j,i),j=1,4)
920      format(7X,I1,4X,F10.8,3X,F9.5,3X,F9.7,2X,
     +               F10.8,1X,4(1X,A2))
         ELSE
         Print *, 'Number of variables not found;',
     +               'input file in wrong format!'
         ENDIF
         ENDIF
         ENDIF
         ENDIF

         minmse=10000
         minsp =10000
         mincp =10000
         ptrmse=0
         ptrcp =0
```

```
                ptrsp =0
                do 30 j= 1,4
                  if(mse(check(j)).lt.minmse) then
                    minmse=mse(check(j))
                    ptrmse=check(j)
                  endif
                  if(sp(check(j)).lt.minsp) then
                    minsp=sp(check(j))
                    ptrsp=check(j)
                  endif
                  if(cp(check(j)).lt.mincp) then
                    mincp=cp(check(j))
                    ptrcp=check(j)
                  endif


30      continue
10      continue
40      continue

             varsmse=varsmse+num(ptrmse)
             varssp =varssp +num(ptrsp)
             varscp =varscp +num(ptrcp)



          do 70 n=1,num(ptrmse)
             if(m(n,ptrmse).EQ 'E1') then
             emse=emse+1
             endif
 70        continue

          do 80 n=1,num(ptrsp)
             if(m(n,ptrsp).eq.'E1') then
             esp=esp+1
             endif
80      continue

            do 90 n=1,num(ptrcp)
              if(m(n,ptrcp).eq.'E1') then
              ecp=ecp+1
              endif
90          continue

             cumemse=cumemse+emse
             cumesp=cumesp+esp
             cumecp=cumecp+ecp
             cmse=num(ptrmse)-emse
             ccp=num(ptrcp)-ecp
             csp=num(ptrsp)-esp
             chartmse(cmse,emse)=chartmse(cmse,emse)+1
             chartcp(ccp,ecp)=chartcp(ccp,ecp)+1
```

```
            chartsp(csp,esp)=chartsp(csp,esp)+1


          write(12,*) 'MSE',num(ptrmse),mse(ptrmse)
     +      ,Sp(ptrmse),cp(ptrmse),' ',
     +       (m(j,ptrmse),j=1,num(ptrmse))
           write(12,*) 'Sp ',num(ptrsp),mse(ptrsp)
     +       ,Sp(ptrsp),cp(ptrsp),' ',
     +       (m(j,ptrsp),j=1,num(ptrsp))
           write(12,*) 'Cp ',num(ptrcp),mse(ptrcp)
     +       ,Sp(ptrcp),cp(ptrcp),' ',
     +       (m(j,ptrcp),j=1,num(ptrcp))
        write(12,*) '******************************
     + ************',
     +              '**********************'
         write(12,*) ' '
         write(12,*) ' '
20       continue
            nummse = real(varsmse)/60.0
            numsp  = real(varssp)/60.0
            numcp  = real(varscp)/60.0
            mseeer= real(cumemse)/60.0
            cpeer = real(cumecp) /60.0
            speer = real(cumesp) /60.0
            msepm = 1-(mseeer/nummse)
            cppm = 1-(cpeer/numcp)
            sppm = 1-(speer/numsp)

          write(12,*) 'The avg number of vars using MSE',
     +                 ' was ', nummse
         write(12,*) 'The avg number of extraneous vars from
     + MSE',
     +                ' was', mseeer
          write(12,*) '****** The PM for MSE was ', msepm,'
     + ******'
          write(12,*) ' '
         write(12,*) 'The avg number of vars using Sp was ',
     +                numsp
         write(12,*) 'The avg number of extraneous vars from
     +  Sp',
     +                ' was ',speer
          write(12,*) '****** The PM for Sp was ', sppm,'
     + ******'
          write(12,*) '  '
          write(12,*) 'The avg number of vars using Cp was',
     +                numcp
          write(12,*) 'The avg number of extraneous vars
     + from Cp',
     +                ' was ',cpeer
          write(12,*) '****** The PM for Cp was ', cppm,'
     + ******'
```

```
            write(12,*) '  '
            write(12,*) '  '
            write(12,*) 'Correct Vars (0-3,   wn) -VS- ',
     +                  'Extraneous Vars (0-3,across)'
            write(12,*) '  '
            write(12,*) ' MSE TABLE'
            write(12,*) '  '
                do 100 i=0,3
                   write(12,*) (chartmse(i,j),j=0,3)

100              continue
                 write(12,*) ' '
                 write(12,*) ' '
                 write(12,*) 'Sp TABLE'
                 write(12,*)
                 do 110 i=0,3
                    write(12,*) (chartsp(i,j),j=0,3)

110               continue

                  write(12,*) ' '
                  write(12,*) ' '
                  write(12,*) 'Cp TABLE'
                  do 120 i=0,3
                    write (12,*) (chartcp(i,j),j=0,3)
120               continue

      write(13,*) (DesignPoint-2),'     ',msepm,'     ',
     +            sppm,'     ',cppm

50    Continue
      Close(11)
      Close(12)
      Close(13)
      GO TO 1200

* Error trap: ******************************************

1000  Continue
      Print 1100, '+++ ERROR WHILE OPENING FILE +++',
     +            '      error code = ', IERROR
1100  FORMAT(/1X, A/ 1X, A, I8/)

**********************************************************

1200  CONTINUE
      Print *,'Counting complete.  ', NewOut,' written.'
      END
```

```
*****************************************************************
**************** FORTRAN PROGRAM COUNT3.FOR ****************
*****************************************************************

      SUBROUTINE COUNT3 (NewOut)
        integer num(63),i,j,k,ptrmse,ptrsp
        integer ptrcp,varsmse,varssp,varscp
        integer check(6)
        integer n,emse,esp,ecp,DesignPoint
        integer ccp,cmse,csp
        integer chartmse(0:3,0:3),chartcp(0:3,0:3)
        integer chartsp(0:3,0:3)
        real MSE(63),Sp(63),cp(63),r2(63)
        real minmse,minsp,mincp,nummse,numcp,numsp
        real mseeer,cpeer,speer
        real msepm,cppm,sppm
        character*2 m(6,63)
        character*20 NewOut
        check(1)=1
        check(2)=7
        check(3)=22
        check(4)=42
        check(5)=57
        check(6)=63
        DesignPoint=2

           do 7 i = 0,3
             do 3 k = 0,3
               chartmse(i,k)=0
               chartcp(i,k)=0
               chartsp(i,k)=0
3            continue
7          continue

        varsmse=0
        varssp =0
        varscp =0
        cumemse=0
        cumesp=0
        cumecp=0

        open (unit=11, file='temp.dat', status='old',
     +       iostat=IERROR, err=1000)
        open (unit=12, file=NewOut, status='new',
     +        iostat=IERROR, err=1000)
        open (unit=13, file='PM3.dat', status='new',
     +        iostat=IERROR, err=1000)
        write(13,*)'    DESIGNPOINT  MSE                ',

     +             'SP                   CP'

        Do 50 jj=2,64,2
```

```
            Write(12,*)' '
            Write(12,*)' '
            Write(12,*)'****************** DESIGN POINT ',
     +                  DesignPoint,' ********************'
            Write(12,*)' '
            DesignPoint=DesignPoint+2

         do 20 k=1,60
         do 10 i=1,63
                 emse=0
                 esp=0
                 ecp=0
                 read(11,'(1X,I1)',end=40)num(i)

                 IF (num(i).EQ.1) THEN
       read(11,905,end=40)num(i),r2(i),cp(i),MSE(i),Sp(i)
     +            ,m(1,i)
905              format(7X,I1,4X,F10.8,3X,F9.5,3X,F9.7,2X,
     +                  F10.8,2X,A2)
                 ELSE
                 IF (num(i).EQ.2) THEN
         read(11,910,end=40)num(i),r2(i),cp(i),MSE(i),Sp(i)

     +            ,(m(j,i),j=1,2)
910              format(7X,I1,4X,F10.8,3X,F9.5,3X,F9.7,2X,
     +                  F10.8,1X,2(1X,A2))
                 ELSE

                 IF (num(i).EQ.3) THEN
         read(11,915,end=40)num(i),r2(i),cp(i),MSE(i),Sp(i)

     +            ,(m(j,i),j=1,3)
915              format(7X,I1,4X,F10.8,3X,F9.5,3X,F9.7,2X,
     +                  F10.8,1X,3(1X,A2))
                 ELSE

                 IF (num(i).EQ.4) THEN
         read(11,920,end=40)num(i),r2(i),cp(i),MSE(i),Sp(i)

     +            ,(m(j,i),j=1,4)
920              format(7X,I1,4X,F10.8,3X,F9.5,3X,F9.7,2X,
     +                  F10.8,1X,4(1X,A2))
                 ELSE

                 IF (num(i).EQ.5) THEN
         read(11,925,end=40)num(i),r2(i),cp(i),MSE(i),Sp(i)

     +            ,(m(j,i),j=1,5)
925              format(7X,I1,4X,F10.8,3X,F9.5,3X,F9.7,2X,
     +                  F10.8,1X,5(1X,A2))
                 ELSE
```

```
                IF (num(i).EQ.6) THEN
            read(11,930,end=40)num(i),r2(i),cp(i),MSE(i),Sp(i)
     +              ,(m(j,i),j=1,6)
930                 format(7X,I1,4X,F10.8,3X,F9.5,3X,F9.7,2X,
     +                      F10.8,1X,6(1X,A2))
                    ELSE
                    Print *, 'Number of variables not found;',
     +                      'input file in wrong format!'
                    ENDIF
                    ENDIF
                    ENDIF
                    ENDIF
                    ENDIF
                    ENDIF

                    minmse=10000
                    minsp =10000
                    mincp =10000
                    ptrmse=0
                    ptrcp =0
                    ptrsp =0
                    do 30 j= 1,6
                      if(mse(check(j)).lt.minmse) then
                        minmse=mse(check(j))
                        ptrmse=check(j)
                      endif
                      if(sp(check(j)).lt.minsp) then
                        minsp=sp(check(j))
                        ptrsp=check(j)
                      endif
                      if(cp(check(j)).lt.mincp) then
                        mincp=cp(check(j))
                        ptrcp=check(j)
                      endif


30          continue
10          continue
40          continue

                varsmse=varsmse+num(ptrmse)
                varssp =varssp +num(ptrsp)
                varscp =varscp +num(ptrcp)

              do 70 n=1,num(ptrmse)
                if(m(n,ptrmse).EQ.'E1') then
                emse=emse+1
                elseif(m(n,ptrmse).eq.'E2') then
                emse=emse+1
                elseif(m(n,ptrmse).eq.'E3') then
                emse=emse+1
```

```
            else
               continue
             endif

70         continue
          do 80 n=1,num(ptrsp)
             if(m(n,ptrsp).eq.'E1') then
             esp=esp+1
             elseif(m(n,ptrsp).eq.'E2') then
             esp=esp+1
             elseif(m(n,ptrsp).eq.'E3') then
             esp=esp+1
             else
             continue
             endif

80          continue
            do 90 n=1,num(ptrcp)
              if(m(n,ptrcp).eq.'E1') then
              ecp=ecp+1
              elseif(m(n,ptrcp).eq.'E2') then
              ecp=ecp+1
              elseif(m(n,ptrcp).eq.'E3') then
              ecp=ecp+1
              else
              continue
              endif

90            continue
             cumemse=cumemse+emse
             cumesp=cumesp+esp
             cumecp=cumecp+ecp
             cmse=num(ptrmse)-emse
             ccp=num(ptrcp)-ecp
             csp=num(ptrsp)-esp
             chartmse(cmse,emse)=chartmse(cmse,emse)+1
             chartcp(ccp,ecp)=chartcp(ccp,ecp)+1
             chartsp(csp,esp)=chartsp(csp,esp)+1



           write(12,*) 'MSE',num(ptrmse),mse(ptrmse)
     +       ,Sp(ptrmse),cp(ptrmse),' ',
     +        (m(j,ptrmse),j=1,num(ptrmse))
           write(12,*) 'Sp ',num(ptrsp),mse(ptrsp)
     +        ,Sp(ptrsp),cp(ptrsp),' ',
     +        (m(j,ptrsp),j=1,num(ptrsp))
           write(12,*) 'Cp ',num(ptrcp),mse(ptrcp)
     +        ,Sp(ptrcp),cp(ptrcp),' ',
     +        (m(j,ptrcp),j=1,num(ptrcp))
      write(12,*) '****************************************',
     +            '**********************'
```

```
      write(12,*) ' '
      write(12,*) ' '
20    continue
         nummse = real(varsmse)/60.0
         numsp  = real(varssp)/60.0
         numcp  = real(varscp)/60.0
         mseeer= real(cumemse)/60.0
         cpeer = real(cumecp) /60.0
         speer = real(cumesp) /60.0
         msepm = 1-(mseeer/nummse)
         cppm = 1-(cpeer/numcp)
         sppm = 1-(speer/numsp)

      write(12,*) 'The avg number of vars using MSE',
     +                  ' was ', nummse
      write(12,*) 'The avg number of extraneous vars
     + from MSE',
     +                  ' was ', mseeer
      write(12,*)'*** The PM for MSE was ',msepm,' ****'
      write(12,*) ' '
      write(12,*) 'The avg number of vars using Sp was',
     +                  numsp
      write(12,*) 'The avg number of extraneous vars
     +from Sp',
     +                  ' was ',speer
      write(12,*)'****** The PM for Sp was ',sppm,' ***'

      write(12,*) '  '
      write(12,*) 'The avg number of vars using Cp was',
     +                  numcp
      write(12,*) 'The avg number of extraneous vars
     + from Cp',
     +                  ' was ',cpeer
      write(12,*)'****** The PM for Cp was ',cppm,' ***'
      write(12,*) '  '
      write(12,*) '  '
      write(12,*) 'Correct Vars (0-3,down) -VS- ',
     +                 'Extraneous Vars (0-3,across)'
      write(12,*) '  '
      write(12,*) ' MSE TABLE'
      write(12,*) '  '
          do 100 i=0,3
             write(12,*) (chartmse(i,j),j=0,3)

100        continue
           write(12,*) ' '
           write(12,*) ' '
           write(12,*) 'Sp TABLE'
           write(12,*)
           do 110 i=0,3
              write(12,*) (chartsp(i,j),j=0,3)
```

```
110             continue

                write(12,*) ' '
                write(12,*) ' '
                write(12,*) 'Cp TABLE'
                do 120 i=0,3
                  write (12,*) (chartcp(i,j),j=0,3)
120             continue
      write(13,*)(DesignPoint-2),'     ',msepm,
     +           '     ',sppm,'     ',cppm

50    Continue

      Close(11)
      Close(12)
      Close(13)
      GO TO 1200

* Error trap: ******************************************

1000  Continue
      Print 1100, '+++ ERROR WHILE OPENING FILE +++',
     +            '      error code = ', IERROR
1100  FORMAT(/1X, A/ 1X, A, I8/)

************************************************************

1200  CONTINUE
      Print *,'Counting complete.  ', NewOut,' written.'
      END
```

```
************************************************************
                      FILCOUNT.FOR
*This program takes SAS R-Squared listings in any file  and
*extracts models with the lowest MSE, Cp, and Sp and then
*figures the performance measure (PM).  This program calls
*subroutines Count1.for and Count3.for and write the *calcu-
lated PM's to *PM1.dat and PM3.dat, respectively.
************************************************************

      Character*20 NewIn
      Character*20 NewOut
      Character*80 Line
      CHARACTER  I, J
      Integer Var
      Logical VarFlag

5     Continue
      Print *,'Name of file to examine? (20 char or less;',
     +        ' "*" to quit)'
      Read (*,'(A20)') NewIn
      If (NewIn(1:1).EQ.'*') GO TO 999
      Print *,'Output file? (20 char or less)'
      Read (*,'(A20)') NewOut
7     Continue
      Print *,'Number of extraneous variables? (1 or 3
     +ONLY!!)'
      Read (*, '(I1)') Var
      If ((Var.NE.1).AND.(Var.NE.3)) GO TO 7

9     Continue
      VarFlag = (Var.EQ.3)

      Open (unit=10, file=NewIn, status='OLD',
     &      iostat=IERROR, err=1000)

      Open (unit=11, file='temp.dat', status='NEW',
     &      iostat=IERROR, err=1000)

10    Continue
      Read(10,200,END=888) Line
      I = LINE (8:8)
      J = LINE (9:11)

      IF (VarFlag) GO TO 777

      IF ((I.EQ.'1').AND.(J.EQ.'   ')) THEN
        WRITE (11,201) I
        WRITE (11,200) LINE
      ELSE

        IF ((I.EQ.'2').AND.(J.EQ.'   ')) THEN
          WRITE (11,201) I
```

```
          WRITE (11,200) LINE
        ELSE

          IF ((I.EQ.'3').AND.(J.EQ.'    ')) THEN
            WRITE (11,201) I
            WRITE (11,200) LINE
          ELSE

            IF ((I.EQ.'4').AND.(J.EQ.'    ')) THEN
              WRITE (11,201) I
              WRITE (11,200) LINE
            ENDIF

          ENDIF

        ENDIF

      ENDIF

      GO TO 10

777   Continue
      IF ((I.EQ.'1').AND.(J.EQ.'    ')) THEN
        WRITE (11,201) I
        WRITE (11,200) LINE
      ELSE

        IF ((I.EQ.'2').AND.(J.EQ.'    ')) THEN
          WRITE (11,201) I
          WRITE (11,200) LINE
        ELSE

          IF ((I.EQ.'3').AND.(J.EQ.'    ')) THEN
            WRITE (11,201) I
            WRITE (11,200) LINE
          ELSE

            IF ((I.EQ.'4').AND.(J.EQ.'    ')) THEN
              WRITE (11,201) I
              WRITE (11,200) LINE
            ELSE

              IF ((I.EQ.'5').AND.(J.EQ.'    ')) THEN
                WRITE (11,201) I
                WRITE (11,200) LINE
              ELSE

                IF ((I.EQ.'6').AND.(J.EQ.'    ')) THEN
                  WRITE (11,201) I
                  WRITE (11,200) LINE
                ENDIF
```

```
              ENDIF

            ENDIF

          ENDIF

        ENDIF

      ENDIF

      GO TO 10


200   Format (A80)
201   Format (1X,A1)
888   Continue
      Close(10)
      Close(11)

      Print *,'Filtering complete on ',NewIn,'.  Counting
     +begun.'

      IF (VarFlag) THEN
        Call Count3(NewOut)
        Print *,'Counting complete.  PM"s calculated for',
     +          ' designpoints with 3 extraneous variables',
     +          ' and written to PM3.dat.'
        Print *,' '
      ELSE
        Call Count1(NewOut)
        Print *,'Counting complete.  PM"s calculated for',
     +          ' designpoints with 1 extraneous variable',
     +          ' and written to PM1.dat.'
        Print *,' '
      ENDIF
      GO TO 5

999   Continue

      Print *,'Processing complete. Program terminated.'
      Stop

* Error trap: ******************************************
1000  Continue
      Print 1100, '+++ ERROR WHILE OPENING FILE +++',
     &            '      error code = ', IERROR
1100  FORMAT(/1X, A/ 1X, A, I8/)
      GO TO 5
************************************************************
      END
```

```
*****************************************************************
                        FILSTEPCOUNT.FOR
*This program takes SAS Forward Selection Stepwise listings
*in any file  and extracts models according to Miller's
*Method and then figures the performance measure (PM).  The
*program reads 1 extraneous variable data files from
*Step1_Input.dat and 3 extranceous variables data files from
*Step3_Input.dat, forms a temporary file called TEMP.DAT and
*then calls subroutines StepCount1.for and StepCount3.for to
*analysis the data.
*****************************************************************

      Character*14 NewIn
      Character*20 NewOut
      Character*80 Line
      Character*1 I, J
      Character*2 K
      Integer Var
      Logical VarFlag, BatchFlag

5     Continue
      Print *,'Interactive(I) or Batch(B) mode? (I or B
     +only):'
      Read (*,'(A1)')Mode
      IF ((Mode.NE.'I').AND.(Mode.NE.'B')) Go to 5
      BatchFlag=(Mode.EQ.'B')
      IF (BatchFlag) Go to 6
      Print *,'Name of file to examine? (20 char or less;',
     +        ' "*" to quit)'
      Read (*,'(A20)') NewIn
      If (NewIn(1:1).EQ.'*') GO TO 999
6     Continue
      Print *,'Output file? (20 char or less)'
      Read (*,'(A20)') NewOut
7     Continue
      Print *,'Number of extraneous variables? (1 or 3
     +ONLY!!)'
      Read (*, '(I1)') Var
      If ((Var.NE.1).AND.(Var.NE.3)) GO TO 7

9     Continue
      VarFlag = (Var.EQ.3)

      IF ((VarFlag).AND.(BatchFlag)) THEN
        Open (unit=9, file='Step3_Input.dat', status='OLD',
     +        iostat=IERROR, err=1000)

      ELSE
        IF ((.NOT.VarFlag).AND.(BatchFlag)) THEN
          Open (unit=9, file='Step1_Input.dat',
     + status='OLD',
```

```
     +          iostat=IERROR, err=1000)

        ENDIF
       ENDIF
11     Continue
       IF (LatchFlag) Read(9,'(A14)',END=666) NewIn
       Print *,'Filtering begun on ',NewIn,'.Filtered data ',
     +        'is being dumped to TEMP.DAT.'

       Open (unit=10, file=NewIn, status='OLD',
     &       iostat=IERROR, err=1000)

       Open (unit=11, file='temp.dat', status='NEW',
     &       iostat=IERROR, err=1000)

10     Continue
       Read(10,200,END=888) Line
       I = LINE (5:5)
       J = LINE (6:8)
       K = LINE (4:5)

       IF (VarFlag) GO TO 777

       IF ((I.EQ.'1').AND.(J.EQ.'   ')) THEN
         WRITE (11,200) LINE
       ELSE

        IF ((I.EQ.'2').AND.(J.EQ.'   ')) THEN
         WRITE (11,200) LINE
        ELSE

         IF ((I.EQ.'3').AND.(J.EQ.'   ')) THEN
          WRITE (11,200) LINE
         ELSE

          IF ((I.EQ.'4').AND.(J.EQ.'   ')) THEN
           WRITE (11,200) LINE
          ELSE

           IF ((I.EQ.'5').AND.(J.EQ.'   ')) THEN
            WRITE (11,200) LINE
           ELSE

            IF ((I.EQ.'6').AND.(J.EQ.'   ')) THEN
             WRITE (11,200) LINE
            ELSE

             IF ((I.EQ.'7').AND.(J.EQ.'   ')) THEN
              WRITE (11,200) LINE
             ELSE

              IF ((I.EQ.'8').AND.(J.EQ.'   ')) THEN
```

```
            WRITE (11,200) LINE
           ENDIF

          ENDIF

         ENDIF

        ENDIF

       ENDIF

      ENDIF

     ENDIF

     GO TO 10

777  Continue
     IF ((I.EQ.'1').AND.(J.EQ.'    ')) THEN
       WRITE (11,200) LINE
     ELSE

      IF ((I.EQ.'2').AND.(J.EQ.'    ')) THEN
       WRITE (11,200) LINE
      ELSE

       IF ((I.EQ.'3').AND.(J.EQ.'    ')) THEN
        WRITE (11,200) LINE
       ELSE

        IF ((I.EQ.'4').AND.(J.EQ.'    ')) THEN
         WRITE (11,200) LINE
        ELSE

         IF ((I.EQ.'5').AND.(J.EQ.'    ')) THEN
          WRITE (11,200) LINE
         ELSE

          IF ((I.EQ.'6').AND.(J.EQ.'    ')) THEN
           WRITE (11,200) LINE
          ELSE

           IF ((I.EQ.'7').AND.(J.EQ.'    ')) THEN
            WRITE (11,200) LINE
           ELSE

            IF ((I.EQ.'8').AND.(J.EQ.'    ')) THEN
             WRITE (11,200) LINE
            ELSE
```

```
                IF ((I.EQ.'9').AND.(J.EQ.'    ')) THEN
                 WRITE (11,200) LINE
                ELSE

                 IF ((K.EQ.'10').AND.(J.EQ.'    ')) THEN
                  WRITE (11,200) LINE
                 ELSE

                  IF ((K.EQ.'11').AND.(J.EQ.'    ')) THEN
                   WRITE (11,200) LINE
                  ELSE

                   IF ((K.EQ.'12').AND.(J.EQ.'    ')) THEN
                    WRITE (11,200) LINE
                   ENDIF

                  ENDIF

                 ENDIF

                ENDIF

               ENDIF

              ENDIF

             ENDIF

            ENDIF

           ENDIF

          ENDIF

         ENDIF

        ENDIF

        GO TO 10


200     Format (A80)

666     Continue
        BatchFlag=.FALSE.

888     Continue
        IF (BatchFlag) THEN
          Close(10)
          Go to 11
        ELSE
          Close(9)
```

```
        Close(10)
        Close(11)
       ENDIF

       Print *,'Filtering complete.  Analysis of data in',
      +        ' TEMP.DAT has begun.',' Analysis results
      +will',
      +        ' be dumped to ',Newout,'.'

       IF (VarFlag) THEN
         Call Stepcount3(NewOut)
       ELSE
         Call Stepcount1(NewOut)
       ENDIF
       GO TO 5

999    Continue

       Print *,'Processing complete. Program terminated.'
       Stop

* Error trap: ********************************************
1000   Continue
       Print 1100, '+++ ERROR WHILE OPENING FILE +++',
      &            '      error code = ', IERROR
1100   FORMAT(/1X, A/ 1X, A, I8/)
       GO TO 5
*************************************************************
       END
```

```
*****************************************************************
*************** FORTRAN PROGRAM MILLBETA.FOR ***************
*****************************************************************

      Logical ErrFlag
      CALL BETA1
      CALL BETA3

      ErrFlag = .FALSE.

      Call MILLTM1(ErrFlag)
      If (ErrFlag) Go to 999
      Print *,'TMSEP''s calculated for designpoints with',
     +        ' 1 extraneous variables and written to
     + MILLTM1.DAT.'
      Print *,' '

      Call MILLTM3(ErrFlag)
      If (ErrFlag) Go to 999
      Print *,'TMSEP''s calculated for designpoints with',
     +        ' 3 extraneous variables and written to
     + MILLTM3.DAT.'
      Print *,' '

999   Continue

      Print *,'Processing complete. Program terminated.'

      STOP
      END
```

```
*****************************************************************
*************** FORTRAN PROGRAM MILLSAS.FOR ***************
*****************************************************************

      INTEGER VarInModel, DP, REP, ActualREP

      CHARACTER*3 J
      CHARACTER*6 filename
      CHARACTER*11 Model1ex
      CHARACTER*17 Model3ex
      CHARACTER*80 LINE


      OPEN (unit=10, file='Miller1Beta.sas', status='NEW',
     +      iostat=IERROR, err=1400)
      OPEN (unit=11, file='Miller3Beta.sas', status='NEW',
     +      iostat=IERROR, err=1400)
      OPEN (unit=12, file='Step1_all.dat', status='OLD',
     +      iostat=IERROR, err=1400)
      OPEN (unit=13, file='Step3_all.dat', status='OLD',
     +      iostat=IERROR, err=1400)



      Do 60 DP=1,64

      VarInModel = 0
      ActualREP = 0
      filename = '      '
      Model1ex = '          '
      Model3ex = '                '

      IF (DP.EQ.1) THEN
        filename = '01.dat'
        GO TO 10
      ENDIF

      IF (DP.EQ.2) THEN
        filename = '02.dat'
        GO TO 20
      ENDIF

      IF (DP.EQ.3) THEN
        filename = '03.dat'
        GO TO 10
      ENDIF

      IF (DP.EQ.4) THEN
        filename = '04.dat'
        GO TO 20
      ENDIF
```

```
IF (DP.EQ.5) THEN
  filename = '05.dat'
  GO TO 10
ENDIF

IF (DP.EQ.6) THEN
  filename = '06.dat'
  GO TO 20
ENDIF

IF (DP.EQ.7) THEN
  filename = '07.dat'
  GO TO 10
ENDIF

IF (DP.EQ.8) THEN
  filename = '08.dat'
  GO TO 20
ENDIF

IF (DP.EQ.9) THEN
  filename = '09.dat'
  GO TO 10
ENDIF

IF (DP.EQ.10) THEN
  filename = '10.dat'
  GO TO 20
ENDIF

IF (DP.EQ.11) THEN
  filename = '11.dat'
  GO TO 10
ENDIF

IF (DP.EQ.12) THEN
  filename = '12.dat'
  GO TO 20
ENDIF

IF (DP.EQ.13) THEN
  filename = '13.dat'
  GO TO 10
ENDIF

IF (DP.EQ.14) THEN
  filename = '14.dat'
  GO TO 20
ENDIF

IF (DP.EQ.15) THEN
  filename = '15.dat'
```

```
  GO TO 10
ENDIF

IF (DP.EQ.16) THEN
  filename = '16.dat'
  GO TO 20
ENDIF

IF (DP.EQ.17) THEN
  filename = '17.dat'
  GO TO 10
ENDIF

IF (DP.EQ.18) THEN
  filename = '18.dat'
  GO TO 20
ENDIF

IF (DP.EQ.19) THEN
  filename = '19.dat'
  GO TO 10
ENDIF

IF (DP.EQ.20) THEN
  filename = '20.dat'
  GO TO 20
ENDIF

IF (DP.EQ.21) THEN
  filename = '21.dat'
  GO TO 10
ENDIF

IF (DP.EQ.22) THEN
  filename = '22.dat'
  GO TO 20
ENDIF

IF (DP.EQ.23) THEN
  filename = '23.dat'
  GO TO 10
ENDIF

IF (DP.EQ.24) THEN
  filename = '24.dat'
  GO TO 20
ENDIF

IF (DP.EQ.25) THEN
  filename = '25.dat'
  GO TO 10
ENDIF
```

```
IF (DP.EQ.26) THEN
  filename = '26.dat'
  GO TO 20
ENDIF

IF (DP.EQ.27) THEN
  filename = '27.dat'
  GO TO 10
ENDIF

IF (DP.EQ.28) THEN
  filename = '28.dat'
  GO TO 20
ENDIF

IF (DP.EQ.29) THEN
  filename = '29.dat'
  GO TO 10
ENDIF

IF (DP.EQ.30) THEN
  filename = '30.dat'
  GO TO 20
ENDIF

IF (DP.EQ.31) THEN
  filename = '31.dat'
  GO TO 10
ENDIF

IF (DP.EQ.32) THEN
  filename = '32.dat'
  GO TO 20
ENDIF

IF (DP.EQ.33) THEN
  filename = '33.dat'
  GO TO 10
ENDIF

IF (DP.EQ.34) THEN
  filename = '34.dat'
  GO TO 20
ENDIF

IF (DP.EQ.35) THEN
  filename = '35.dat'
  GO TO 10
ENDIF

IF (DP.EQ.36) THEN
  filename = '36.dat'
```

```
  GO TO 20
ENDIF

IF (DP.EQ.37) THEN
  filename = '37.dat'
  GO TO 10
ENDIF

IF (DP.EQ.38) THEN
  filename = '38.dat'
  GO TO 20
ENDIF

IF (DP.EQ.39) THEN
  filename = '39.dat'
  GO TO 10
ENDIF

IF (DP.EQ.40) THEN
  filename = '40.dat'
  GO TO 20
ENDIF

IF (DP.EQ.41) THEN
  filename = '41.dat'
  GO TO 10
ENDIF

IF (DP.EQ.42) THEN
  filename = '42.dat'
  GO TO 20
ENDIF

IF (DP.EQ.43) THEN
  filename = '43.dat'
  GO TO 10
ENDIF

IF (DP.EQ.44) THEN
  filename = '44.dat'
  GO TO 20
ENDIF

IF (DP.EQ.45) THEN
  filename = '45.dat'
  GO TO 10
ENDIF

IF (DP.EQ.46) THEN
  filename = '46.dat'
  GO TO 20
ENDIF
```

```
IF (DP.EQ.47) THEN
  filename = '47.dat'
  GO TO 10
ENDIF

IF (DP.EQ.48) THEN
  filename = '48.dat'
  GO TO 20
ENDIF

IF (DP.EQ.49) THEN
  filename = '49.dat'
  GO TO 10
ENDIF

IF (DP.EQ.50) THEN
  filename = '50.dat'
  GO TO 20
ENDIF

IF (DP.EQ.51) THEN
  filename = '51.dat'
  GO TO 10
ENDIF

IF (DP.EQ.52) THEN
  filename = '52.dat'
  GO TO 20
ENDIF

IF (DP.EQ.53) THEN
  filename = '53.dat'
  GO TO 10
ENDIF

IF (DP.EQ.54) THEN
  filename = '54.dat'
  GO TO 20
ENDIF

IF (DP.EQ.55) THEN
  filename = '55.dat'
  GO TO 10
ENDIF

IF (DP.EQ.56) THEN
  filename = '56.dat'
  GO TO 20
ENDIF

IF (DP.EQ.57) THEN
  filename = '57.dat'
```

```
        GO TO 10
      ENDIF

      IF (DP.EQ.58) THEN
        filename = '58.dat'
        GO TO 20
      ENDIF

      IF (DP.EQ.59) THEN
        filename = '59.dat'
        GO TO 10
      ENDIF

      IF (DP.EQ.60) THEN
        filename = '60.dat'
        GO TO 20
      ENDIF

      IF (DP.EQ.61) THEN
        filename = '61.dat'
        GO TO 10
      ENDIF

      IF (DP.EQ.62) THEN
        filename = '62.dat'
        GO TO 20
      ENDIF

      IF (DP.EQ.63) THEN
        filename = '63.dat'
        GO TO 10
      ENDIF

      IF (DP.EQ.64) THEN
        filename = '64.dat'
        GO TO 20
      ENDIF

10    CONTINUE

924   CONTINUE
      READ (12,925,END=1350) LINE
925   FORMAT(1X,A80)
      J = LINE(1:3)
      IF (J.NE.'Rep') GO TO 924

      DO 30 REP=1,60

      READ (12,900,END=1000) ActualREP, VarInModel, Modellex
900   FORMAT(5X,I2,12X,I1,6X,A11)

      IF (REP.NE.ActualREP) GO TO 1100
```

```
      WRITE (10,901) FILENAME
901   FORMAT (1X,'FILENAME NEW ''',A6,''';')

      WRITE (10,902)
902   FORMAT (1X,'DATA NEW;')

      WRITE (10,903)
903   FORMAT (1X,'INFILE NEW;')

      WRITE (10,904)
904   FORMAT (1X,'INPUT SETNUM Y X1 X2 X3 X4 E1;')

      WRITE (10,905) ActualREP
905   FORMAT (1X,'IF SETNUM^=',I2,' THEN DELETE;')


      IF (VarInModel.EQ.0) THEN

        WRITE (10,906)
906     FORMAT (1X,'INTERCEP = 1;')

        WRITE (10,907)
907     FORMAT (1X,'PROC RSQUARE DATA=NEW NOINT B;')

        WRITE (10,908)
908     FORMAT (1X,'MODEL Y = INTERCEP;')

      ELSE

        WRITE (10,909)
909     FORMAT (1X,'PROC RSQUARE DATA=NEW B;')

        WRITE (10,910) Modellex, VarInModel
910     FORMAT (1X,'MODEL Y = ',A11,' /INCLUDE=',I1,';')

      ENDIF

      WRITE (10,*)

30    CONTINUE



      GO TO 50


20    CONTINUE

926   CONTINUE
      READ (13,927,END=1375) LINE
927   FORMAT(1X,A80)
      J = LINE(1:3)
```

```
      IF (J.NE.'Rep') GO TO 926

      DO 40 REP=1,60

      READ (13,911,END=1200) ActualREP, VarInModel, Model3ex
911   FORMAT(5X,I2,12X,I1,6X,A17)

      IF (REP.NE.ActualREP) GO TO 1300

      WRITE (11,912) FILENAME
912   FORMAT (1X,'FILENAME NEW ''',A6,''';')

      WRITE (11,913)
913   FORMAT (1X,'DATA NEW;')

      WRITE (11,914)
914   FORMAT (1X,'INFILE NEW;')

      WRITE (11,915)
915   FORMAT (1X,'INPUT SETNUM Y X1 X2 X3 X4 E1 E2 E3;')

      WRITE (11,916) ActualREP
916   FORMAT (1X,'IF SETNUM^=',I2,' THEN DELETE;')


      IF (VarInModel.EQ.0) THEN

        WRITE (11,917)
917     FORMAT (1X,'INTERCEP = 1;')

        WRITE (11,918)
918     FORMAT (1X,'PROC RSQUARE DATA=NEW NOINT B;')

        WRITE (11,919)
919     FORMAT (1X,'MODEL Y = INTERCEP;')

      ELSE

        WRITE (11,920)
920     FORMAT (1X,'PROC RSQUARE DATA=NEW B;')

        WRITE (11,921) Model3ex, VarInModel
921     FORMAT (1X,'MODEL Y = ',A17,' /INCLUDE=',I1,';')

      ENDIF

      WRITE (11,*)

40    CONTINUE

50    CONTINUE
```

```
60    CONTINUE

      CLOSE (10)
      CLOSE (11)
      CLOSE (12)
      CLOSE (13)

      PRINT *, 'Program completed successfully with ',
     +         DP-1,' designpoints and ',REP-1,
     +         'replications.'

      GO TO 1500

* ERROR TRAP************************************************

1000  CONTINUE

      PRINT *,'ERROR WHILE READING STEP1_ALL.DAT.',
     +         'UNEXPECTED END OF FILE.'
      GO TO 1500

1100  CONTINUE

      PRINT *,'STEP1_ALL.DAT IN UNEXPECTED FORMAT.',
     +         'REP COUNTER DOES NOT AGREE WITH FILE.'
      GO TO 1500

1200  CONTINUE

      PRINT *,'ERROR WHILE READING STEP3_ALL.DAT.',
     +         'UNEXPECTED END OF FILE.'
      GO TO 1500

1300  CONTINUE

      PRINT *,'STEP3_ALL.DAT IN UNEXPECTED FORMAT.',
     +         'REP COUNTER DOES NOT AGREE WITH FILE.'
      GO TO 1500

1350  CONTINUE

      PRINT *,'STEP1_ALL.DAT IN UNEXPECTED FORMAT.',
     +         'DP COUNTER DOES NOT AGREE WITH FILE.'

1375  CONTINUE

      PRINT *,'STEP3_ALL.DAT IN UNEXPECTED FORMAT.',
     +         'DP COUNTER DOES NOT AGREE WITH FILE.'

1400  CONTINUE
      PRINT 1401, '+++ ERROR WHILE OPENING FILE +++',
     +            '       error code = ', IERROR
```

```
1401    FORMAT (/1X, A/ 1X, A, I8/)

*****************************************************************

1500    CONTINUE
        STOP
        END
```

```
************************************************************
*                FORTRAN PROGRAM MILLTM1.FOR
*
* This program is designed to take the 64 groups of 60
* models selected via Miller's method and the corresponding
* 3840 data sets and find the "real" MSEP for each of the 32
* odd designpoints of 64 design points.
************************************************************

      Subroutine MILLTM1(ErrFlag)

      Integer h,i,j,k,p,r,s
      Integer num

      Real b0, betas(4)
      Real x(4,20),x3ex1(4,20),ex,y
      Real ypredmillers
      Real ymsepmillers
      Real yssepmillers
      Real sumyssepmillers
      Real sumdifmillers
      Real dpymsepmillers

      Character*6 Infile

      Logical ErrFlag
      Open(unit=11,file= 'MILLER1BETA.DAT',status='old',
     +                    iostat=IERROR, err=1000)
      Open (unit=13,file='MILLTM1.DAT',status='new',
     +      iostat=IERROR, err=1002)

      Write (13,902)
902   Format (1X,'TMSEPs calculated for the Miller''s
     + method:')

      Write (13,901)
901   Format (1X,'DP',9X,'Miller''s')

      Do 5 r=1,63,2

       If (r.EQ.1) then
        Infile='01.dat'
       Else
        If (r.EQ.3) then
         Infile='03.dat'
        Else
         If (r.EQ.5) then
          Infile='05.dat'
         Else
          If (r.EQ.7) then
           Infile='07.dat'
          Else
```

```
If (r.EQ.9) then
 Infile='09.dat'
Else
 If (r.EQ.11) then
  Infile='11.dat'
 Else
  If (r.EQ.13) then
   Infile='13.dat'
  Else
   If (r.EQ.15) then
    Infile='15.dat'
   Else
    If (r.EQ.17) then
     Infile='17.dat'
    Else
     If (r.EQ.19) then
      Infile='19.dat'
     Else
      If (r.EQ.21) then
       Infile='21.dat'
      Else
       If (r.EQ.23) then
        Infile='23.dat'
       Else
        If (r.EQ.25) then
         Infile='25.dat'
        Else
         If (r.EQ.27) then
          Infile='27.dat'
         Else
          If (r.EQ.29) then
           Infile='29.dat'
          Else
           If (r.EQ.31) then
            Infile='31.dat'
           Else
            If (r.EQ.33) then
             Infile='33.dat'
            Else
             If (r.EQ.35) then
              Infile='35.dat'
             Else
              If (r.EQ.37) then
               Infile='37.dat'
              Else
               If (r.EQ.39) then
                Infile='39.dat'
               Else
                If (r.EQ.41) then
                 Infile='41.dat'
                Else
                 If (r.EQ.43) then
```

```
 Infile='43.dat'
Else
 If (r.EQ.45) then
  Infile='45.dat'
 Else
  If (r.EQ.47) then
   Infile='47.dat'
  Else
   If (r.EQ.49) then
    Infile='49.dat'
   Else
    If (r.EQ.51) then
     Infile='51.dat'
    Else
     If (r.EQ.53) then
      Infile='53.dat'
     Else
      If (r.EQ.55) then
       Infile='55.dat'
      Else
       If (r.EQ.57) then
        Infile='57.dat'
       Else
        If (r.EQ.59) then
         Infile='59.dat'
        Else
         If (r.EQ.61) then
          Infile='61.dat'
         Else
          If (r.EQ.63) then
           Infile='63.dat'
          Endif
         Endif
        Endif
       Endif
      Endif
     Endif
    Endif
   Endif
  Endif
 Endif
Endif
```

```
                Endif
               Endif
              Endif
             Endif
            Endif
           Endif
          Endif
         Endif
        Endif
       Endif
      Endif
      Endif
      Open(unit=12,file=Infile,status='old',iostat=IERROR,
     +                    err=1001)

      sumyssepmillers = 0
      sumdifmillers = 0

      Do 20 k=1,60

       yssepmillers=0

        Read (11,*,end=1003)  num, b0, betas(1),
     +        betas(2), betas(3), betas(4)


       If(((r.GE.17).AND.(r.LE.32)).OR.(r.GE.49)) then
        s = 20
       Else
        s = 10
       Endif

       Do 50 h= 1,s

       Read(12,*,end=1004)set,y,x(1,h),x(2,h),x(3,h),
     +  x(4,h),ex

        ypredmillers= b0
        yactual = 0
        x3ex1(1,h)= x(1,h)
        x3ex1(2,h)= x(2,h)
        x3ex1(3,h)= x(3,h)
        x3ex1(4,h)= ex

        Do 60 p=1,4
         yactual = yactual+x(p,h)
60      Continue

        Do 70 p=1,4
         ypredmillers = ypredmillers + betas(p)*x3ex1(p,h)
70      Continue

        yssepmillers = ((ypredmillers-yactual)**real(2))
```

```
     +                    + yssepmillers

50        Continue

          sumyssepmillers = sumyssepmillers + yssepmillers
          sumdifmillers = sumdifmillers + (s-num)

20        Continue

          dpymsepmillers  = sumyssepmillers / sumdifmillers

          Write(13,900) r, dpymsepmillers
900       Format (1X,I2,5X,F10.6)

          Close (12)

5         Continue

          Close (11)
          Close (13)
          Go to 1300
******Error trap**********'*********************************


1000  Print *,'Something''s wrong with MILLER1BETA.DAT.'
      Go to 1100
1001  Print *,'Something''s wrong with ',Infile
      Go to 1100
1002  Print *,'Can''t seem to create MILLTM1.DAT.'
      Go to 1100
1003  Print *, 'MILLER1BETA.DAT in unexpected format.'
      ErrFlag = .TRUE.
      Go to 1300
1004  Print *, 'File ',Infile,' is in an unexpected format.'
      ErrFlag = .TRUE.
      Go to 1300

1100  Continue
      Print 1200, '+++ ERROR WHILE OPENING FILE +++',
     +            '      error code = ',IERROR
1200  Format (/1X, A/ 1X, A, I8/)
      ErrFlag = .TRUE.
*************************************************************

1300  Continue

      END
```

```
*****************************************************************
*
*                    FORTRAN PROGRAM MILLTM3.FOR
*
* This program is designed to take the 64 groups of 60
* models selected via Miller's method and the corresponding
* 3840 data sets and find the "real" MSEP for each of the 32
* even design points of 64 design points.
*****************************************************************

      Subroutine MILLTM3(ErrFlag)

      Integer h,i,j,k,p,r,s
      Integer num

      Real b0, betas(6)
      Real x(4,20),x3ex3(6,20),y,ex1,ex2,ex3
      Real ypredmillers
      Real ymsepmillers
      Real yssepmillers
      Real sumyssepmillers
      Real sumdifmillers
      Real dpymsepmillers

      Character*6 Infile

      Logical ErrFlag

      Open(unit=11,file= 'MILLER3BETA.DAT',status='old',
     +                    iostat=IERRCR,err=1000)
     + Open(unit=13,file='MILLTM3.DAT',status='new',
     + iostat=IERROR, err=1002)

      Write (13,902)
902   Format (1X,'TMSEPs calculated for Miller''s method:')

      Write (13,901)
901   Format (1X,'DP',9X,'Miller''s')

      Do 5 r=2,64,2

       If (r.EQ.2) then
        Infile='02.dat'
       Else
        If (r.EQ.4) then
         Infile='04.dat'
        Else
         If (r.EQ.6) then
          Infile='06.dat'
         Else
          If (r.EQ.8) then
           Infile='08.dat'
```

```
Else
 If (r.EQ.10) then
  Infile='10.dat'
 Else
  If (r.EQ.12) then
   Infile='12.dat'
  Else
   If (r.EQ.14) then
    Infile='14.dat'
   Else
    If (r.EQ.16) then
     Infile='16.dat'
    Else
     If (r.EQ.18) then
      Infile='18.dat'
     Else
      If (r.EQ.20) then
       Infile='20.dat'
      Else
       If (r.EQ.22) then
        Infile='22.dat'
       Else
        If (r.EQ.24) then
         Infile='24.dat'
        Else
         If (r.EQ.26) then
          Infile='26.dat'
         Else
          If (r.EQ.28) then
           Infile='28.dat'
          Else
           If (r.EQ.30) then
            Infile='30.dat'
           Else
            If (r.EQ.32) then
             Infile='32.dat'
            Else
             If (r.EQ.34) then
              Infile='34.dat'
             Else
              If (r.EQ.36) then
               Infile='36.dat'
              Else
               If (r.EQ.38) then
                Infile='38.dat'
               Else
                If (r.EQ.40) then
                 Infile='40.dat'
                Else
                 If (r.EQ.42) then
                  Infile='42.dat'
                 Else
```

```
If (r.EQ.44) then
 Infile='44.dat'
Else
 If (r.EQ.46) then
  Infile='46.dat'
 Else
  If (r.EQ.48) then
   Infile='48.dat'
  Else
   If (r.EQ.50) then
    Infile='50.dat'
   Else
    If (r.EQ.52) then
     Infile='52.dat
    Else
     If (r.EQ.54) then
      Infile='54.dat'
     Else
      If (r.EQ.56) then
       Infile='56.dat'
      Else
       If (r.EQ.58) then
        Infile='58.dat'
       Else
        If (r.EQ.60) then
         Infile='60.dat'
        Else
         If (r.EQ.62) then
          Infile='62.dat'
         Else
          If (r.EQ.64) then
           Infile='64.dat'
          Endif
         Endif
        Endif
       Endif
      Endif
     Endif
    Endif
   Endif
  Endif
 Endif
Endif
```

```
                  Endif
                 Endif
                Endif
               Endif
              Endif
             Endif
            Endif
           Endif
          Endif
         Endif
        Endif
       Endif
       Open(unit=12,file=Infile,status='old',iostat=IERROR,
     +                      err=1001)

       sumyssepmillers = 0
       sumdifmillers = 0

       Do 20 k=1,60

        yssepmillers=0

      Read (11,*,end=1003)num,b0,betas(1),betas(2),betas(3),
     +                             betas(4), betas(5), betas(6)


       If(((r.GE.17).AND.(r.LE.32)).OR.(r.GE.49)) then
        s = 20
       Else
        s = 10
       Endif

       Do 50 h= 1,s

        Read(12,*,end=1004)set,y,x(1,h),x(2,h),
     +    x(3,h),x(4,h),ex1,ex2,ex3

        ypredmillers = b0
        yactual = 0
        x3ex3(1,h)= x(1,h)
        x3ex3(2,h)= x(2,h)
        x3ex3(3,h)= x(3,h)
        x3ex3(4,h)= ex1
        x3ex3(5,h)= ex2
        x3ex3(6,h)= ex3

        Do 60 p=1,4
         yactual = yactual+x(p,h)
60      Continue

        Do 70 p=1,6
         ypredmillers = ypredmillers + betas(p)*x3ex3(p,h)
```

```
70          Continue

            yssepmillers = ((ypredmillers-yactual)**real(2))
     +                        + yssepmillers

50          Continue

            sumyssepmillers = sumyssepmillers + yssepmillers
            sumdifmillers = sumdifmillers + (s-num)

20          Continue

            dpymsepmillers = sumyssepmillers / sumdifmillers

            Write(13,900) r, dpymsepmillers
900         Format (1X,I2,5X,F10.6)

            Close (12)

5           Continue

            Close (11)
            Close (13)
            Go to 1300
******Error trap*************************************************


1000  Print *,'Something''s wrong with MILLER3BETA.DAT.'
      Go to 1100
1001  Print *,'Something''s wrong with ',Infile
      Go to 1100
1002  Print *,'Can''t seem to create MILLTM3.DAT.'
      Go to 1100
1003  Print *, 'MILLER3BETA.DAT in unexpected format.'
      ErrFlag = .TRUE.
      Go to 1300
1004  Print *, 'File ',Infile,' is in an unexpected format.'
      ErrFlag = .TRUE.
      Go to 1300

1100  Continue
      Print 1200, '+++ ERROR WHILE OPENING FILE +++',
     +            '      error code = ',IERROR
1200  Format (/1X, A/ 1X, A, I8/)
      ErrFlag = .TRUE.
*****************************************************************
1300  Continue

      END
```

```
************************************************************
************** FORTRAN PROGRAM STEPCOUNT1.FOR **************
************************************************************

      SUBROUTINE Stepcount1 (NewOut)

        integer num, numvar,h,i,j,k,n,p,q,r,s,t,v,w,x,y,z
        integer emiller, varsmiller, cumemiller, cmiller
        integer chartmiller(0:3,0:3), ReadCount
        real avgvars, avgevars, millerpm
        character*1 numchar
        character*2 m(9), Model_var, Good_model(5)
        character*20 NewOut
        logical ModelNotFound, EndofFile

        num=0
        emiller=0
        varsmiller=0
        cumemiller=0
        cmiller=0
        ReadCount=0
        ModelNotFound=.TRUE.
        EndofFile=.FALSE.


        do 10 i=1,8
           m(i)='  '
10      continue

        do 20 j=1,4
           Good_model(j)='  '
20      continue

        do 30 k=0,3
          do 40 h=0,3
            chartmiller(k,h)=0
40        continue
30      continue

        open (unit=11, file='temp.dat', status='old',
     +       iostat=IERROR, err=1000)
        open (unit=12, file=NewOut, status='new',
     +        iostat=IERROR, err=1000)
        open (unit=13, file='PMstep1.dat', status='new',
     +        iostat=IERROR, err=1000)
        write(13,*)'     DESIGNPOINT  MILLER''S PM'


        ReadCount=ReadCount+1
           Read(11,900,end=90)numchar, Model_var
900        Format(4X,A1,4X,A2)
           IF (numchar.EQ.'1') THEN
```

```
          num=1
        ELSE
         IF (numchar.EQ.'2') THEN
           num=2
         ELSE
          IF (numchar.EQ.'3') THEN
            num=3
          ELSE
           IF (numchar.EQ.'4') THEN
             num=4
           ELSE
            IF (numchar.EQ.'5') THEN
              num=5
            ELSE
             IF (numchar.EQ.'6') THEN
               num=6
             ELSE
              IF (numchar.EQ.'7') THEN
                num=7
              ELSE
               IF (numchar.EQ.'8') THEN
                 num=8
               ELSE
               Print *, 'Unexpected format in TEMP.DAT: ',
+                       'Numbers 1,2,3, etc., not found!'
                Go to 1300
               ENDIF
              ENDIF
             ENDIF
            ENDIF
           ENDIF
          ENDIF
         ENDIF
        ENDIF

        IF (num.NE.1) THEN
        Print *,'Processing terminated.  Input file ',
+           'in unexpected format: 1st number must be 1.'
          Go to 1300
        ELSE
          m(num)=Model_var
        ENDIF

      Do 60 n=1,63,2
        Write(12,*)' '
        Write(12,*)' '
        Write(12,*)' '
        Write(12,*) '********************* DESIGN POINT ',
+                   n,' *********************'
        Write (12,*)' '
        Write(12,*)'Replication    #Vars    Model'
```

```
          do 70 p=1,60

80          Continue
            IF (EndofFile) THEN
          Print *,'Unexpected file format!  File does not ',
     +           'have correct # of design points and reps.'
             Go to 1300
            ENDIF
            ReadCount=ReadCount+1
            Read(11,900,end=90) numchar, Model_var

             IF (numchar.EQ.'1') THEN
               num=1
             ELSE
              IF(numchar.EQ.'2') THEN
                num=2
              ELSE
               IF (numchar.EQ.'3') THEN
                 num=3
               ELSE
                IF (numchar.EQ.'4') THEN
                  num=4
                ELSE
                 IF (numchar.EQ.'5') THEN
                   num=5
                 ELSE
                  IF (numchar.EQ.'6') THEN
                    num=6
                  ELSE
                   IF (numchar.EQ.'7') THEN
                     num=7
                   ELSE
                    IF (numchar.EQ.'8') THEN
                      num=8
                    ELSE
                   Print *, 'Unexpected format in TEMP.DAT: ',
     +                      'Numbers 1,2,3, etc., not found!'
                     Go to 1300
                    ENDIF
                   ENDIF
                  ENDIF
                 ENDIF
                ENDIF
               ENDIF
              ENDIF
             ENDIF
            ENDIF

110         Continue
            IF (num.NE.1) THEN
                 m(num)=Model_var
            ELSE
                 continue
```

```
                do 100 q=1,4
              IF ((m(q)(1:1).NE.'R').AND.ModelNotFound) THEN
                    Good_model(q)=m(q)
                    numvar=numvar+1
                  ELSE
                    ModelNotFound=.FALSE.
                  ENDIF
100               continue
              Write(12,901)p, numvar, (Good_model(r),r=1,4)
901               FORMAT (' ',4X,I2,11X,I2,6X,A2,1X,A2,1X,
     +                     A2,1X,A2)
                  Go to 120
              ENDIF
              GO TO 80

120       continue
          IF (numvar.LE.0) Go to 140
          do 130 s=1,numvar
              if(m(s).EQ.'E1') then
               emiller=emiller+1
              endif
130       continue

140          varsmiller=varsmiller+numvar
             cumemiller=cumemiller+emiller
             cmiller=numvar-emiller
             chartmiller(cmiller,emiller)=chartmiller(cmiller,
     +                   emiller)+1


         do 150 t=1,8
            m(t)='  '
150      Continue

         m(num)=Model_var

         do 160 v=1,4
            Good_model(v)='  '
160      continue

         emiller=0
         numvar=0
         ModelNotFound=.TRUE.

70         continue

           write(12,*) ' '
           write(12,*) '****************************************',
     +                 '****************************************'
           write(12,*) ' '

           IF (varsmiller.GT.0) THEN
```

```
          avgvars = real(varsmiller)/60.0
          avgevars = real(cumemiller)/60.0
          millerpm = 1-(avgevars/avgvars)
        ELSE
          avgvars=0
          avgevars=0
          millerpm=0
        ENDIF

        write(12,*) 'The avg number of vars using
     +  Miller''s',
     +              ' method was ', avgvars
        write(12,*) 'The avg number of extraneous vars
     + from',
     +              ' Miller''s method was', avgevars
        write(12,*) '****** The PM for Miller''s was ',
     +               millerpm,' ******'
        write(12,*) ' '
        write(12,*) '  '
        write(12,*) 'Correct Vars (0-3, down) -VS- ',
     +                'Extraneous Vars (0-3,across)'
        write(12,*) '  '
        write(12,*) ' Table for Miller''s Method'
        write(12,*) '  '
              do 170 w=0,3
                  write(12,*) (chartmiller(w,x),x=0,3)

170             continue
        Write(13,*) n,'     ',millerpm

        do 180 y = 0,3
          do 190 z = 0,3
            chartmiller(y,z)=0
190       continue
180     continue

        varsmiller=0
        cumemiller=0

60    Continue
      Close(11)
      Close(12)
      Close(13)
      GO TO 1200

* Error trap: ****************************************

1000  Continue
      Print 1100, '+++ ERROR WHILE OPENING FILE +++',
     +            '      error code = ', IERROR
1100  FORMAT(/1X, A/ 1X, A, I8/)
```

```
************************************************************

1200  CONTINUE
      Print *,'Counting complete.  ', NewOut,' written.'
      Go to 1300

90    Print*,'End of File encountered at line ',ReadCount
      Print*,'Design Point:',n,'  Replication:',p
      num=1
      Model_var='**'
      EndofFile=.TRUE.
      Go to 110

1300  CONTINUE
      END
```

```
*****************************************************************
************** FORTRAN PROGRAM STEPCOUNT3.FOR **************
*****************************************************************

      SUBROUTINE Stepcount3 (NewOut)

        integer num, numvar,h,i,j,k,n,p,q,r,s,t,v,w,x,y,z
        integer emiller, varsmiller, cumemiller, cmiller
        integer chartmiller(0:3,0:3), ReadCount
        real avgvars, avgevars, millerpm
        character*2 m(13), Model_var, Good_model(7), numchar
        character*20 NewOut
        logical ModelNotFound, EndofFile

        num=0
        emiller=0
        varsmiller=0
        cumemiller=0
        cmiller=0
        ReadCount=0
        ModelNotFound=.TRUE.
        EndofFile=.FALSE.


        do 10 i=1,12
           m(i)='  '
10      continue

        do 20 j=1,6
           Good_model(j)='  '
20      continue

        do 30 k=0,3
          do 40 h=0,3
            chartmiller(k,h)=0
40        continue
30      continue

        open (unit=11, file='temp.dat', status='old',
     +       iostat=IERROR, err=1000)
        open (unit=12, file=NewOut, status='new',
     +        iostat=IERROR, err=1000)
        open (unit=13, file='PMstep3.dat', status='new',
     +        iostat=IERROR, err=1000)
        write(13,*)'     DESIGNPOINT  MILLER''S PM'


        ReadCount=ReadCount+1
           Read(11,900,end=90)numchar, Model_var
900        Format(3X,A2,4X,A2)
           IF (numchar.EQ.' 1') THEN
             num=1
```

```
ELSE
 IF (numchar.EQ.' 2') THEN
   num=2
 ELSE
  IF (numchar.EQ.' 3') THEN
    num=3
  ELSE
   IF (numchar.EQ.' 4') THEN
     num=4
   ELSE
    IF (numchar.EQ.' 5') THEN
      num=5
    ELSE
     IF (numchar.EQ.' 6') THEN
       num=6
     ELSE
      IF (numchar.EQ.' 7') THEN
        num=7
      ELSE
       IF (numchar.EQ.' 8') THEN
         num=8
       ELSE
        IF (numchar.EQ.' 9') THEN
          num=9
        ELSE
         IF (numchar.EQ.'10') THEN
           num=10
         ELSE
          IF (numchar.EQ.'11') THEN
            num=11
          ELSE
           IF (numchar.EQ.'12') THEN
             num=12
           ELSE
       Print *, 'Unexpected format in TEMP.DAT: ',
+                'Numbers 1,2,3, etc., not found!'
             Go to 1300
           ENDIF
          ENDIF
         ENDIF
        ENDIF
       ENDIF
      ENDIF
     ENDIF
    ENDIF
   ENDIF
  ENDIF
 ENDIF
ENDIF

IF (num.NE.1) THEN
  Print *,'Processing terminated.  Input file ',
```

```
     +          'in unexpected format: 1st number must be 1.'
                Go to 1300
               ELSE
                 m(num)=Model_var
               ENDIF

            Do 60 n=2,64,2
               Write(12,*)' '
               Write(12,*)' '
               Write(12,*)' '
               Write(12,*) '********************* DESIGN POINT ',
     +                      n,' *********************'
               Write (12,*)' '
               Write(12,*)'Replication    #Vars    Model'

               do 70 p=1,50

     80          Continue
                 IF (EndofFile) THEN
                Print *,'Unexpected file format!  File does not ',
     +          'have the correct # of design points and reps.'
                  Go to 1300
                 ENDIF
                 ReadCount=ReadCount+1
                 Read(11,900,end=90) numchar, Model_var

                  IF (numchar.EQ.' 1') THEN
                    num=1
                  ELSE
                   IF(numchar.EQ.' 2') THEN
                     num=2
                   ELSE
                    IF (numchar.EQ.' 3') THEN
                      num=3
                    ELSE
                     IF (numchar.EQ.' 4') THEN
                       num=4
                     ELSE
                      IF (numchar.EQ.' 5') THEN
                        num=5
                      ELSE
                       IF (numchar.EQ.' 6') THEN
                         num=6
                       ELSE
                        IF (numchar.EQ.' 7') THEN
                          num=7
                        ELSE
                         IF (numchar.EQ.' 8') THEN
                           num=8
                         ELSE
                          IF (numchar.EQ.' 9') THEN
                            num=9
```

```
                    ELSE
                     IF (numchar.EQ.'10') THEN
                       num=10
                     ELSE
                      IF (numchar.EQ.'11') THEN
                        num=11
                      ELSE
                       IF (numchar.EQ.'12') THEN
                         num=12
                       ELSE
              Print *, 'Unexpected format in TEMP.DAT: ',
     +                  'Numbers 1,2,3, etc., not found!'
                        Go to 1300
                       ENDIF
                      ENDIF
                     ENDIF
                    ENDIF
                   ENDIF
                  ENDIF
                 ENDIF
                ENDIF
               ENDIF
              ENDIF
             ENDIF

110           Continue
              IF (num.NE.1) THEN
                   m(num)=Model_var
              ELSE
                   continue
                   do 100 q=1,6
             IF ((m(q)(1:1).NE.'R').AND.ModelNotFound) THEN
                    Good_model(q)=m(q)
                    numvar=numvar+1
                  ELSE
                    ModelNotFound=.FALSE.
                  ENDIF
100               continue
             Write(12,901)p, numvar, (Good_model(r),r=1,6)
901               FORMAT (' ',4X,I2,11X,I2,6X,A2,1X,A2,1X,
     +                     A2,1X,A2,1X,A2,1X,A2)
                  Go to 120
             ENDIF
             GO TO 80

120     continue
        IF (numvar.LE.0) Go to 140
        do 130 s=1,numvar
            if((m(s).EQ.'E1').OR.(m(s).EQ.'E2').OR.
     +         (m(s).EQ.'E3')) then
        emiller=emiller+1
```

```
            endif
130       continue

140           varsmiller=varsmiller+numvar
              cumemiller=cumemiller+emiller
              cmiller=numvar-emiller
              chartmiller(cmiller,emiller)=chartmiller(cmiller,
     +                    emiller)+1


          do 150 t=1,12
             m(t)='  '
150       Continue

          m(num)=Model_var

          do 160 v=1,6
             Good_model(v)='  '
160       continue

          emiller=0
          numvar=0
          ModelNotFound=.TRUE.

70          continue

            write(12,*) ' '
            write(12,*) '****************************************',
     +                  '****************************************'
            write(12,*) ' '

            IF (varsmiller.GT.0) THEN
              avgvars = real(varsmiller)/60.0
              avgevars = real(cumemiller)/60.0
              millerpm = 1-(avgevars/avgvars)
            ELSE
              avgvars=0
              avgevars=0
              millerpm=0
            ENDIF

          write(12,*) 'The avg number of vars using Miller''s',
     +                ' method was ', avgvars
          write(12,*) 'The avg number of extraneous vars from',
     +                ' Miller''s method was', avgevars
            write(12,*) '****** The PM for Miller''s was ',
     +                  millerpm,' ******'
            write(12,*) ' '
            write(12,*) '  '
            write(12,*) 'Correct Vars (0-3, down) -VS- ',
     +                  'Extraneous Vars (0-3,across)'
            write(12,*) '  '
```

```
        write(12,*) ' Table for Miller''s Method'
        write(12,*) '   '
              do 170 w=0,3
                  write(12,*) (chartmiller(w,x),x=0,3)

170             continue
        Write(13,*) n,'      ',millerpm

        do 180 y = 0,3
          do 190 z = 0,3
            chartmiller(y,z)=0
190       continue
180     continue

        varsmiller=0
        cumemiller=0

60    Continue
      Close(11)
      Close(12)
      Close(13)
      GO TO 1200

* Error trap: ********************************************

1000  Continue
      Print 1100, '+++ ERROR WHILE OPENING FILE +++',
     +            '      error code = ', IERROR
1100  FORMAT(/1X, A/ 1X, A, I6/)

************************************************************

1200  CONTINUE
      Print *,'Counting complete.  ', NewOut,' written.'
      Go to 1300

90    Print*,'End of File encountered at line ',ReadCount
      Print*,'Design Point:',n,'  Replication:',p
      num=1
      Model_var='**'
      EndofFile=.TRUE.
      Go to 110

1300  CONTINUE
      END
```

```
************************************************************
*                          TMSEP.FOR                       *
* This program takes SAS R-Squared listings in any file
* (with switches MSE, SP, CP, and B) and extracts models
* with the lowest MSE, Cp, and Sp.  Then, using the original
* data files (01.dat, 02.dat,...,64.dat), it calculates the
* theoretical performance measure (TMSEP).  This program
* calls subroutines  * * TMSEP1.FOR, TMSEP2.FOR, TMSEP3.FOR,
* and TMSEP4.FOR and writes the calculated TMSEP's to
* TMSEP.DAT.
************************************************************

      Character*20 NewIn
      Character*132 Line
      CHARACTER  I, J, K, L
      Integer Var
      Logical VarFlag,ErrFlag

5     Continue
      Print *,'Name of file to examine? (20 char or less;',
     +        ' "*" to quit)'
      Read (*,'(A20)') NewIn
      If (NewIn(1:1).EQ.'*') GO TO 999

7     Continue
      Print *,'Number of extraneous variables? (1 or 3
     +ONLY!!)'
      Read (*,'(I1)') Var
      If ((Var.NE.1).AND.(Var.NE.3)) Go To 7

      VarFlag = (Var.EQ.3)


      Open (unit=10, file=NewIn, status='OLD',
     &      iostat=IERROR, err=1000)

      Open (unit=11, file='temp.dat', status='NEW',
     &      iostat=IERROR, err=1000)

10    Continue
      Read(10,200,END=888) Line
      I = LINE (14:14)
      J = LINE (15:17)
      K = LINE (9:9)
      L = LINE (10:12)

      IF (VarFlag) GO TO 777

      IF ((I.EQ.'1').AND.(J.EQ.'   ')) THEN
        WRITE (11,200) LINE
      ELSE
```

```
          IF ((J.EQ.'2').AND.(J.EQ.'    ')) THEN
            WRITE (11,200) LINE
          ELSE

            IF ((I.EQ.'3').AND.(J.EQ.'    ')) THEN
              WRITE (11,200) LINE
            ELSE

              IF ((I.EQ.'4').AND.(J.EQ.'    ')) THEN
                WRITE (11,200) LINE
              ENDIF

            ENDIF

          ENDIF

        ENDIF

        GO TO 10

777     Continue
        IF ((K.EQ.'1').AND.(L.EQ.'    ')) THEN
          WRITE (11,200) LINE
        ELSE

          IF ((K.EQ.'2').AND.(L.EQ.'    ')) THEN
            WRITE (11,200) LINE
          ELSE

            IF ((K.EQ.'3').AND.(L.EQ.'    ')) THEN
              WRITE (11,200) LINE
            ELSE

              IF ((K.EQ.'4').AND.(L.EQ.'    ')) THEN
                WRITE (11,200) LINE
              ELSE

                IF ((K.EQ.'5').AND.(L.EQ.'    ')) THEN
                  WRITE (11,200) LINE
                ELSE

                  IF ((K.EQ.'6').AND.(L.EQ.'    ')) THEN
                    WRITE (11,200) LINE
                  ENDIF

                ENDIF

              ENDIF

            ENDIF

          ENDIF

        ENDIF
```

```
      ENDIF

      GO TO 10


200   Format (A132)
888   Continue
      Close(10)
      Close(11)

      Print *,'Filtering complete on ',NewIn,'.',
     +        '  TMSEP calculations begun.'

      ErrFlag = .FALSE.

      IF (VarFlag) THEN
        Call TMSEP3(ErrFlag)
        If (ErrFlag) Go to 5
        Print *,'TMSEP''s calculated for designpoints with',

     +  ' 3 extraneous variables and written to TMSEP3.DAT.'
        Print *,' '
      ELSE
        Call TMSEP1(ErrFlag)
        If (ErrFlag) Go to 5
        Print *,'TMSEP''s calculated for designpoints with',

     +          ' 1 extraneous variables and written to
TMSEP1.DAT.'
        Print *,' '
      ENDIF
      GO TO 5

999   Continue

      Print *,'Processing complete. Program terminated.'
      Stop

* Error trap: ********************************************
1000  Continue
      Print 1100, '+++ ERROR WHILE OPENING FILE +++',
     &            '      error code = ', IERROR
1100  FORMAT(/1X, A/ 1X, A, I8/)
      GO TO 5
*************************************************************
      END
```

```
************************************************************
*                  FORTRAN PROGRAM TMSEP1.FOR
*
* This program is designed to take a modifed SAS progam and
* an existing data set and find the "real" MSEP for the
* models chosen by mse, sp, and cp criteria.
************************************************************

      Subroutine TMSEP1(ErrFlag)

      Integer h,i,j,k,p,r,s,ptrmse,ptrsp,ptrcp
      Integer check(4),num(15)

      Real b0(15),r2(15),cp(15),mse(15),sp(15),betas(4,15)
      Real x(4,20),x3ex1(4,20),ex,y
      Real minmse,mincp,minsp
      Real ypredcp,ypredmse,ypredsp
      Real ymsepmse,ymsepsp,ymsepcp
      Real yssepmse,yssepsp,yssepcp
      Real sumyssepmse,sumyssepsp,sumyssepcp
      Real sumdifmse,sumdifsp,sumdifcp
      Real dpymsepmse,dpymsepsp,dpymsepcp

      Character*6 Infile

      Logical ErrFlag
      check(1)=1
      check(2)=5
      check(3)=11
      check(4)=15

      Open(unit=11,file= 'TEMP.DAT',status='old',
     + iostat=IERROR,err=1000)
      Open (unit=13,file='TMSEP1.DAT',status='new',
     + iostat=IERROR,err=1002)

      Write (13,902)
902   Format (1X,'TMSEPs calculated for the following
     + methods:')

      Write (13,901)
901   Format (1X,'DP',9X,'MSE',13X,'SP',13X,'CP')

      Do 5 r=1,63,2

       If (r.EQ.1) then
        Infile='01.dat'
       Else
        If (r.EQ.3) then
         Infile='03.dat'
        Else
         If (r.EQ.5) then
```

```
 Infile='05.dat'
Else
 If (r.EQ.7) then
  Infile='07.dat'
 Else
  If (r.EQ.9) then
   Infile='09.dat'
  Else
   If (r.EQ.11) then
    Infile='11.dat'
   Else
    If (r.EQ.13) then
     Infile='13.dat'
    Else
     If (r.EQ.15) then
      Infile='15.dat'
     Else
      If (r.EQ.17) then
       Infile='17.dat'
      Else
       If (r.EQ.19) then
        Infile='19.dat'
       Else
        If (r.EQ.21) then
         Infile='21.dat'
        Else
         If (r.EQ.23) then
          Infile='23.dat'
         Else
          If (r.EQ.25) then
           Infile='25.dat'
          Else
           If (r.EQ.27) then
            Infile='27.dat'
           Else
            If (r.EQ.29) then
             Infile='29.dat'
            Else
             If (r.EQ.31) then
              Infile='31.dat'
             Else
              If (r.EQ.33) then
               Infile='33.dat'
              Else
               If (r.EQ.35) then
                Infile='35.dat'
               Else
                If (r.EQ.37) then
                 Infile='37.dat'
                Else
                 If (r.EQ.39) then
                  Infile='39.dat'
```

```
Else
 If (r.EQ.41) then
  Infile='41.dat'
 Else
  If (r.EQ.43) then
   Infile='43.dat'
  Else
   If (r.EQ.45) then
    Infile='45.dat'
   Else
    If (r.EQ.47) then
     Infile='47.dat'
    Else
     If (r.EQ.49) then
      Infile='49.dat'
     Else
      If (r.EQ.51) then
       Infile='51.dat'
      Else
       If (r.EQ.53) then
        Infile='53.dat'
       Else
        If (r.EQ.55) then
         Infile='55.dat'
        Else
         If (r.EQ.57) then
          Infile='57.dat'
         Else
          If (r.EQ.59) then
           Infile='59.dat'
          Else
           If (r.EQ.61) then
            Infile='61.dat'
           Else
            If (r.EQ.63) then
             Infile='63.dat'
            Endif
           Endif
          Endif
         Endif
        Endif
       Endif
      Endif
     Endif
    Endif
   Endif
  Endif
 Endif
Endif
```

```
                    Endif
                   Endif
                  Endif
                 Endif
                Endif
               Endif
              Endif
             Endif
            Endif
           Endif
          Endif
         Endif
        Endif
       Endif
      Endif
     Endif
    Endif
    Open(unit=12,file=Infile,status='old',iostat=IERROR,
   +                     err=1001)

    sumyssepmse = 0
    sumyssepsp = 0
    sumyssepcp = 0
    sumdifmse = 0
    sumdifsp = 0
    sumdifcp = 0

    Do 20 k=1,60

     minmse = 10000
     minsp  = 10000
     mincp  = 10000
     yssepmse=0
     yssepsp =0
     yssepcp =0
     ptrmse = 0
     ptrsp  = 0
     ptrcp  = 0

     Do 10 i=1,15
      Read (11,*,end=1003)  num(i), r2(i), cp(1),
   +   mse(i), sp(i), b0(i), betas(1,i), betas(2,i),
   +   betas(3,i), betas(4,i)
10    Continue

     Do 30 j=1,4
      If(mse(check(j)).lt.minmse) then
       minmse = mse(check(j))
       ptrmse = check(j)
      Endif
      If (sp(check(j)).lt.minsp) then
       minsp  = sp(check(j))
       ptrsp  = check(j)
```

```
            Endif
            If (cp(check(j)).lt.mincp) then
             mincp = cp(check(j))
             ptrcp = check(j)
            Endif
30          Continue

            If(((r.GE.17).AND.(r.LE.32)).OR.(r.GE.49)) then
             s = 20
            Else
             s = 10
            Endif

            Do 50 h= 1,s

             Read (12,*,end=1004) set,y,x(1,h),x(2,h),x(3,h),
     +       x(4,h),ex

             ypredmse= b0(ptrmse)
             ypredsp = b0(ptrsp)
             ypredcp = b0(ptrcp)
             yactual = 0
             x3ex1(1,h)= x(1,h)
             x3ex1(2,h)= x(2,h)
             x3ex1(3,h)= x(3,h)
             x3ex1(4,h)= ex

             Do 60 p=1,4
              yactual = yactual+x(p,h)
60           Continue

             Do 70 p=1,4
              ypredmse = ypredmse + betas(p,ptrmse)*x3ex1(p,h)
              ypredsp  = ypredsp  + betas(p,ptrsp) *x3ex1(p,h)
              ypredcp  = ypredcp  + betas(p,ptrcp) *x3ex1(p,h)
70           Continue

             yssepmse = ((ypredmse-yactual)**real(2)) +
     +       yssepmse
             yssepsp  = ((ypredsp -yactual)**real(2)) + yssepsp
             yssepcp  = ((ypredcp -yactual)**real(2)) + yssepcp

50          Continue

            sumyssepmse = sumyssepmse + yssepmse
            sumyssepsp = sumyssepsp + yssepsp
            sumyssepcp = sumyssepcp + yssepcp
            sumdifmse = sumdifmse + (s-num(ptrmse))
            sumdifsp = sumdifsp + (s-num(ptrsp))
            sumdifcp = sumdifcp + (s-num(ptrcp))

20          Continue
```

```
        dpymsepmse  = sumyssepmse / sumdifmse
        dpymsepsp   = sumyssepsp  / sumdifsp
        dpymsepcp   = sumyssepcp  / sumdifcp

        Write(13,900) r, dpymsepmse, dpymsepsp, dpymsepcp
900     Format (1X,I2,5X,F10.6,5X,F10.6,5X,F10.6)

        Close (12)

5       Continue

        Close (11)
        Close (13)
        Go to 1300
******Error trap**********************************************


1000  Print *,'Something''s wrong with TEMP.DAT.'
      Go to 1100
1001  Print *,'Something''s wrong with ',Infile
      Go to 1100
1002  Print *,'Can''t seem to create TMSEP1.DAT.'
      Go to 1100
1003  Print *, 'TEMP.DAT in unexpected format.'
      Go to 1300
1004  Print *, 'File ',Infile,' is in an unexpected format.'
      Go to 1300

1100  Continue
      Print 1200, '+++ ERROR WHILE OPENING FILE +++',
     +            '       error code = ',IERROR
1200  Format (/1X, A/ 1X, A, I8)
      ErrFlag = .TRUE.
*************************************************************
1300  Continue

      END
```

```
*****************************************************************
*                    FORTRAN PROGRAM TMSEP3.FOR
*
* This program is designed to take a modifed SAS progam and
* an existing data set and find the "real" MSEP for the
* models chosen by mse, sp, and cp criteria.
*
*****************************************************************

      Subroutine TMSEP3(ErrFlag)

      Integer h,i,j,k,p,r,s,ptrmse,ptrsp,ptrcp
      Integer check(6),num(63)

      Real b0(63),r2(63),cp(63),mse(63),sp(63),betas(6,63)
      Real x(4,20),x3ex3(6,20),y,ex1,ex2,ex3
      Real minmse,mincp,minsp
      Real ypredcp,ypredmse,ypredsp
      Real ymsepmse,ymsepsp,ymsepcp
      Real yssepmse,yssepsp,yssepcp
      Real sumyssepmse,sumyssepsp,sumyssepcp
      Real sumdifmse,sumdifsp,sumdifcp
      Real dpymsepmse,dpymsepsp,dpymsepcp

      Character*6 Infile

      Logical ErrFlag
      check(1)=1
      check(2)=7
      check(3)=22
      check(4)=42
      check(5)=57
      check(6)=63

     Open(unit=11,file= 'TEMP.DAT',status='old',
    +iostat=IERROR,err=1000)
      Open (unit=13,file='TMSEP3.DAT',status='new',
    +iostat=IERROR,err=1002)

      Write (13,902)
902   Format (1X,'TMSEPs calculated for the following
    + methods:')

      Write (13,901)
901   Format (1X,'DP',9X,'MSE',13X,'SP',13X,'CP')

      Do 5 r=2,64,2

       If (r.EQ.2) then
        Infile='02.dat'
       Else
        If (r.EQ.4) then
```

```
 Infile='04.dat'
Else
 If (r.EQ.6) then
  Infile='06.dat'
 Else
  If (r.EQ.8) then
   Infile='08.dat'
  Else
   If (r.EQ.10) then
    Infile='10.dat'
   Else
    If (r.EQ.12) then
     Infile='12.dat'
    Else
     If (r.EQ.14) then
      Infile='14.dat'
     Else
      If (r.EQ.16) then
       Infile='16.dat'
      Else
       If (r.EQ.18) then
        Infile='18.dat'
       Else
        If (r.EQ.20) then
         Infile='20.dat'
        Else
         If (r.EQ.22) then
          Infile='22.dat'
         Else
          If (r.EQ.24) then
           Infile='24.dat'
          Else
           If (r.EQ.26) then
            Infile='26.dat'
           Else
            If (r.EQ.28) then
             Infile='28.dat'
            Else
             If (r.EQ.30) then
              Infile='30.dat'
             Else
              If (r.EQ.32) then
               Infile='32.dat'
              Else
               If (r.EQ.34) then
                Infile='34.dat'
               Else
                If (r.EQ.36) then
                 Infile='36.dat'
                Else
                 If (r.EQ.38) then
                  Infile='38.dat'
```

```
Else
 If (r.EQ.40) then
  Infile='40.dat'
 Else
  If (r.EQ.42) then
   Infile='42.dat'
  Else
   If (r.EQ.44) then
    Infile='44.dat'
   Else
    If (r.EQ.46) then
     Infile='46.dat'
    Else
     If (r.EQ.48) then
      Infile='48.dat'
     Else
      If (r.EQ.50) then
       Infile='50.dat'
      Else
       If (r.EQ.52) then
        Infile='52.dat'
       Else
        If (r.EQ.54) then
         Infile='54.dat'
        Else
         If (r.EQ.56) then
          Infile='56.dat'
         Else
          If (r.EQ.58) then
           Infile='58.dat'
          Else
           If (r.EQ.60) then
            Infile='60.dat'
           Else
            If (r.EQ.62) then
             Infile='62.dat'
            Else
             If (r.EQ.64) then
              Infile='64.dat'
             Endif
            Endif
           Endif
          Endif
         Endif
        Endif
       Endif
      Endif
     Endif
    Endif
   Endif
  Endif
 Endif
Endif
```

```
                    Endif
                   Endif
                  Endif
                 Endif
                Endif
               Endif
              Endif
             Endif
            Endif
           Endif
          Endif
         Endif
        Endif
       Endif
      Endif
     Endif
    Endif
    Open(unit=12,file=Infile,status='old',iostat=IERROR,
   +                     err=1001)

    sumyssepmse = 0
    sumyssepsp = 0
    sumyssepcp = 0
    sumdifmse = 0
    sumdifsp = 0
    sumdifcp = 0

    Do 20 k=1,60

     minmse = 10000
     minsp  = 10000
     mincp  = 10000
     yssepmse=0
     yssepsp =0
     yssepcp =0
     ptrmse = 0
     ptrsp  = 0
     ptrcp  = 0

     Do 10 i=1,63
      Read (11,*,end=1003)  num(i), r2(i), cp(1),
   +    mse(i), sp(i), b0(i), betas(1,i), betas(2,i),
   +    betas(3,i), betas(4,i),
   +   betas(5,i), betas(6,i)
10    Continue

     Do 30 j=1,6
      If(mse(check(j)).lt.minmse) then
       minmse = mse(check(j))
       ptrmse = check(j)
```

```
            Endif
            If (sp(check(j)).lt.minsp) then
             minsp  = sp(check(j))
             ptrsp  = check(j)
            Endif
            If (cp(check(j)).lt.mincp) then
             mincp = cp(check(j))
             ptrcp = check(j)
            Endif
 30        Continue

           If(((r.GE.17).AND.(r.LE.32)).OR.(r.GE.49)) then
            s = 20
           Else
            s = 10
           Endif

           Do 50 h= 1,s

            Read (12,*,end=1004) set,y,x(1,h),x(2,h),
     +      x(3,h),x(4,h),ex1,ex2,ex3

            ypredmse= b0(ptrmse)
            ypredsp = b0(ptrsp)
            ypredcp = b0(ptrcp)
            yactual = 0
            x3ex3(1,h)= x(1,h)
            x3ex3(2,h)= x(2,h)
            x3ex3(3,h)= x(3,h)
            x3ex3(4,h)= ex1
            x3ex3(5,h)= ex2
            x3ex3(6,h)= ex3

            Do 60 p=1,4
             yactual = yactual+x(p,h)
 60         Continue

            Do 70 p=1,6
             ypredmse = ypredmse + betas(p,ptrmse)*x3ex3(p,h)
             ypredsp  = ypredsp  + betas(p,ptrsp) *x3ex3(p,h)
             ypredcp  = ypredcp  + betas(p,ptrcp) *x3ex3(p,h)
 70         Continue

            yssepmse = ((ypredmse-yactual)**real(2)) +
     +      yssepmse
            yssepsp  = ((ypredsp -yactual)**real(2)) + yssepsp
            yssepcp  = ((ypredcp -yactual)**real(2)) + yssepcp

 50        Continue

           sumyssepmse = sumyssepmse + yssepmse
           sumyssepsp = sumyssepsp + yssepsp
```

```
          sumyssepcp = sumyssepcp + yssepcp
          sumdifmse = sumdifmse + (s-num(ptrmse))
          sumdifsp = sumdifsp + (s-num(ptrsp))
          sumdifcp = sumdifcp + (s-num(ptrcp))

20        Continue

          dpymsepmse  = sumyssepmse / sumdifmse
          dpymsepsp   = sumyssepsp  / sumdifsp
          dpymsepcp   = sumyssepcp  / sumdifcp

          Write(13,900) r, dpymsepmse, dpymsepsp, dpymsepcp
900       Format (1X,I2,5X,F10.6,5X,F10.6,5X,F10.6)

          Close (12)

5         Continue

          Close (11)
          Close (13)
          Go to 1300
******Error trap*************************************************


1000  Print *,'Something''s wrong with TEMP.DAT.'
      Go to 1100
1001  Print *,'Something''s wrong with ',Infile
      Go to 1100
1002  Print *,'Can''t seem to create TMSEP3.DAT.'
      Go to 1100
1003  Print *, 'TEMP.DAT in unexpected format.'
      Go to 1300
1004  Print *, 'File ',Infile,' is in an unexpected format.'
      Go to 1300

1100  Continue
      Print 1200, '+++ ERROR WHILE OPENING FILE +++',
     +            '      error code = ',IERROR
1200  Format (/1X, A/ 1X, A, I8/)
      ErrFlag = .TRUE.
*****************************************************************

1300  Continue

      END
```

Appendix J. SAS Programs

List of SAS Programs

```
;                    SAS Program ERROR1_ALL.SAS
;
option linesize=80;
filename new '01.dat';
data new;
infile new;
input  set y x1 x2 x3 x4 e1  ;
proc rsquare data=new  mse sp cp;
by set;
 model y= x1 x2 x3 e1  ;

filename new '03.dat';
data new;
infile new;
input  set y x1 x2 x3 x4 e1  ;
proc rsquare data=new  mse sp cp;
by set;
 model y= x1 x2 x3 e1  ;

filename new '05.dat';
data new;
infile new;
input  set y x1 x2 x3 x4 e1  ;
proc rsquare data=new  mse sp cp;
by set;
 model y= x1 x2 x3 e1  ;

filename new '07.dat';
data new;
infile new;
input  set y x1 x2 x3 x4 e1  ;
proc rsquare data=new  mse sp cp;
by set;
 model y= x1 x2 x3 e1  ;
.
.
.
.
.
.
.
filename new '63.dat';
data new;
infile new;
input  set y x1 x2 x3 x4 e1  ;
proc rsquare data=new  mse sp cp;
by set;
 model y= x1 x2 x3 e1  ;
```

```
;                    SAS Program ERROR3_ALL.SAS
;
option linesize=80;
filename new '02.dat';
data new;
infile new;
input  set y x1 x2 x3 x4 e1 e2 e3 @;
proc rsquare data=new  mse sp cp;
by set;
 model y= x1 x2 x3 e1 e2 e3;

filename new '04.dat';
data new;
infile new;
input  set y x1 x2 x3 x4 e1 e2 e3 @;
proc rsquare data=new  mse sp cp;
by set;
 model y= x1 x2 x3 e1 e2 e3;

filename new '06.dat';
data new;
infile new;
input  set y x1 x2 x3 x4 e1 e2 e3 @;
proc rsquare data=new  mse sp cp;
by set;
 model y= x1 x2 x3 e1 e2 e3  ;

filename new '08.dat';
data new;
infile new;
input  set y x1 x2 x3 x4 e1 e2 e3 @;
proc rsquare data=new  mse sp cp;
by set;
 model y= x1 x2 x3 e1 e2 e3  ;
.
.
.
.
.
.
.
.
.
filename new '64.dat';
data new;
infile new;
input  set y x1 x2 x3 x4 e1 e2 e3 @;
proc rsquare data=new  mse sp cp;
by set;
 model y= x1 x2 x3 e1 e2 e3  ;
```

```
;
;                SAS Program MILLER1BETA.SAS
;
 FILENAME NEW '01.dat';
 DATA NEW;
 INFILE NEW;
 INPUT SETNUM Y X1 X2 X3 X4 E1;
 IF SETNUM^= 1 THEN DELETE;
 PROC RSQUARE DATA=NEW B;
 MODEL Y = X1 X3 X2     /INCLUDE=3;

 FILENAME NEW '01.dat';
 DATA NEW;
 INFILE NEW;
 INPUT SETNUM Y X1 X2 X3 X4 E1;
 IF SETNUM^= 2 THEN DELETE;
 PROC RSQUARE DATA=NEW B;
 MODEL Y = X1 X3 X2     /INCLUDE=3;

 FILENAME NEW '01.dat';
 DATA NEW;
 INFILE NEW;
 INPUT SETNUM Y X1 X2 X3 X4 E1;
 IF SETNUM^= 3 THEN DELETE;
 PROC RSQUARE DATA=NEW B;
 MODEL Y = X2           /INCLUDE=1;

 FILENAME NEW '01.dat';
 DATA NEW;
 INFILE NEW;
 INPUT SETNUM Y X1 X2 X3 X4 E1;
 IF SETNUM^= 4 THEN DELETE;
 PROC RSQUARE DATA=NEW B;
 MODEL Y = X3           /INCLUDE=1;

 FILENAME NEW '01.dat';
 DATA NEW;
 INFILE NEW;
 INPUT SETNUM Y X1 X2 X3 X4 E1;
 IF SETNUM^= 5 THEN DELETE;
 PROC RSQUARE DATA=NEW B;
 MODEL Y = X1 X3        /INCLUDE=2;

 FILENAME NEW '01.dat';
 DATA NEW;
 INFILE NEW;
 INPUT SETNUM Y X1 X2 X3 X4 E1;
 IF SETNUM^= 6 THEN DELETE;
 PROC RSQUARE DATA=NEW B;
 MODEL Y = X1 X3 E1     /INCLUDE=3;
               .
               .
```

```
.
.
.
.

FILENAME NEW '03.dat';
DATA NEW;
INFILE NEW;
INPUT SETNUM Y X1 X2 X3 X4 E1;
IF SETNUM^= 1 THEN DELETE;
PROC RSQUARE DATA=NEW B;
MODEL Y = X1 X3 X2 E1 /INCLUDE=4;

FILENAME NEW '03.dat';
DATA NEW;
INFILE NEW;
INPUT SETNUM Y X1 X2 X3 X4 E1;
IF SETNUM^= 2 THEN DELETE;
PROC RSQUARE DATA=NEW B;
MODEL Y = X1 X3       /INCLUDE=2;

FILENAME NEW '03.dat';
DATA NEW;
INFILE NEW;
INPUT SETNUM Y X1 X2 X3 X4 E1;
IF SETNUM^= 3 THEN DELETE;
PROC RSQUARE DATA=NEW B;
MODEL Y = X1 X2 X3    /INCLUDE=3;

FILENAME NEW '03.dat';
DATA NEW;
INFILE NEW;
INPUT SETNUM Y X1 X2 X3 X4 E1;
IF SETNUM^= 4 THEN DELETE;
PROC RSQUARE DATA=NEW B;
MODEL Y = X3 X2 X1 E1 /INCLUDE=4;

FILENAME NEW '03.dat';
DATA NEW;
INFILE NEW;
INPUT SETNUM Y X1 X2 X3 X4 E1;
IF SETNUM^= 5 THEN DELETE;
PROC RSQUARE DATA=NEW B;
MODEL Y = X3 X1       /INCLUDE=2;

FILENAME NEW '03.dat';
DATA NEW;
INFILE NEW;
INPUT SETNUM Y X1 X2 X3 X4 E1;
IF SETNUM^= 6 THEN DELETE;
PROC RSQUARE DATA=NEW B;
```

```
MODEL Y = X3 X1 X2     /INCLUDE=3;

                .
                .
                .
                .
                .

FILENAME NEW '63.dat';
DATA NEW;
INFILE NEW;
INPUT SETNUM Y X1 X2 X3 X4 E1;
IF SETNUM^= 1 THEN DELETE;
PROC RSQUARE DATA=NEW B;
MODEL Y = X1 X3 X2 E1 /INCLUDE=4;

FILENAME NEW '63.dat';
DATA NEW;
INFILE NEW;
INPUT SETNUM Y X1 X2 X3 X4 E1;
IF SETNUM^= 2 THEN DELETE;
PROC RSQUARE DATA=NEW B;
MODEL Y = X1 X2 X3     /INCLUDE=3;

FILENAME NEW '63.dat';
DATA NEW;
INFILE NEW;
INPUT SETNUM Y X1 X2 X3 X4 E1;
IF SETNUM^= 3 THEN DELETE;
PROC RSQUARE DATA=NEW B;
MODEL Y = X3 X2 X1 E1 /INCLUDE=4;

FILENAME NEW '63.dat';
DATA NEW;
INFILE NEW;
INPUT SETNUM Y X1 X2 X3 X4 E1;
IF SETNUM^= 4 THEN DELETE;
PROC RSQUARE DATA=NEW B;
MODEL Y = X3 X2 X1     /INCLUDE=3;

FILENAME NEW '63.dat';
DATA NEW;
INFILE NEW;
INPUT SETNUM Y X1 X2 X3 X4 E1;
IF SETNUM^= 5 THEN DELETE;
PROC RSQUARE DATA=NEW B;
MODEL Y = X2 X1 X3     /INCLUDE=3;

                .
                .
                .
                .
                .
```

```
FILENAME NEW '63.dat';
DATA NEW;
INFILE NEW;
INPUT SETNUM Y X1 X2 X3 X4 E1;
IF SETNUM^=60 THEN DELETE;
PROC RSQUARE DATA=NEW B;
MODEL Y = X2 X1 X3     /INCLUDE=3;
```

```
;
;                SAS Program MILLER3BETA.SAS
;
 FILENAME NEW '02.dat';
 DATA NEW;
 INFILE NEW;
 INPUT SETNUM Y X1 X2 X3 X4 E1 E2 E3;
 IF SETNUM^= 1 THEN DELETE;
 INTERCEP = 1;
 PROC RSQUARE DATA=NEW NOINT B;
 MODEL Y = INTERCEP;

 FILENAME NEW '02.dat';
 DATA NEW;
 INFILE NEW;
 INPUT SETNUM Y X1 X2 X3 X4 E1 E2 E3;
 IF SETNUM^= 2 THEN DELETE;
 INTERCEP = 1;
 PROC RSQUARE DATA=NEW NOINT B;
 MODEL Y = INTERCEP;

 FILENAME NEW '02.dat';
 DATA NEW;
 INFILE NEW;
 INPUT SETNUM Y X1 X2 X3 X4 E1 E2 E3;
 IF SETNUM^= 3 THEN DELETE;
 PROC RSQUARE DATA=NEW B;
 MODEL Y = X3 E2 X2           /INCLUDE=3;

 FILENAME NEW '02.dat';
 DATA NEW;
 INFILE NEW;
 INPUT SETNUM Y X1 X2 X3 X4 E1 E2 E3;
 IF SETNUM^= 4 THEN DELETE;
 PROC RSQUARE DATA=NEW B;
 MODEL Y = X3 X2 X1           /INCLUDE=3;

 FILENAME NEW '02.dat';
 DATA NEW;
 INFILE NEW;
 INPUT SETNUM Y X1 X2 X3 X4 E1 E2 E3;
 IF SETNUM^= 5 THEN DELETE;
 PROC RSQUARE DATA=NEW B;
 MODEL Y = X1 X3 E1           /INCLUDE=3;

                .
                .
                .
                .
                .
                .
                .
```

```
FILENAME NEW '04.dat';
DATA NEW;
INFILE NEW;
INPUT SETNUM Y X1 X2 X3 X4 E1 E2 E3;
IF SETNUM^= 1 THEN DELETE;
PROC RSQUARE DATA=NEW B;
MODEL Y = X2 X3 X1              /INCLUDE=3;

FILENAME NEW '04.dat';
DATA NEW;
INFILE NEW;
INPUT SETNUM Y X1 X2 X3 X4 E1 E2 E3;
IF SETNUM^= 2 THEN DELETE;
PROC RSQUARE DATA=NEW B;
MODEL Y = X2 X1 X3              /INCLUDE=3;

FILENAME NEW '04.dat';
DATA NEW;
INFILE NEW;
INPUT SETNUM Y X1 X2 X3 X4 E1 E2 E3;
IF SETNUM^= 3 THEN DELETE;
PROC RSQUARE DATA=NEW B;
MODEL Y = X2 X3                 /INCLUDE=2;

FILENAME NEW '04.dat';
DATA NEW;
INFILE NEW;
INPUT SETNUM Y X1 X2 X3 X4 E1 E2 E3;
IF SETNUM^= 4 THEN DELETE;
PROC RSQUARE DATA=NEW B;
MODEL Y = X3                    /INCLUDE=1;

FILENAME NEW '04.dat';
DATA NEW;
INFILE NEW;
INPUT SETNUM Y X1 X2 X3 X4 E1 E2 E3;
IF SETNUM^= 5 THEN DELETE;
PROC RSQUARE DATA=NEW B;
MODEL Y = X2 X1 X3              /INCLUDE=3;

FILENAME NEW '04.dat';
DATA NEW;
INFILE NEW;
INPUT SETNUM Y X1 X2 X3 X4 E1 E2 E3;
IF SETNUM^= 6 THEN DELETE;
PROC RSQUARE DATA=NEW B;
MODEL Y = X2 X1                 /INCLUDE=2;
```

```
                    .
                    .
FILENAME NEW '64.dat';
DATA NEW;
INFILE NEW;
INPUT SETNUM Y X1 X2 X3 X4 E1 E2 E3;
IF SETNUM^= 1 THEN DELETE;
PROC RSQUARE DATA=NEW B;
MODEL Y = X2 X3 X1           /INCLUDE=3;

FILENAME NEW '64.dat';
DATA NEW;
INFILE NEW;
INPUT SETNUM Y X1 X2 X3 X4 E1 E2 E3;
IF SETNUM^= 2 THEN DELETE;
PROC RSQUARE DATA=NEW B;
MODEL Y = X2 X3 X1 E3 E2     /INCLUDE=5;

FILENAME NEW '64.dat';
DATA NEW;
INFILE NEW;
INPUT SETNUM Y X1 X2 X3 X4 E1 E2 E3;
IF SETNUM^= 3 THEN DELETE;
PROC RSQUARE DATA=NEW B;
MODEL Y = X1 X2 X3           /INCLUDE=3;

FILENAME NEW '64.dat';
DATA NEW;
INFILE NEW;
INPUT SETNUM Y X1 X2 X3 X4 E1 E2 E3;
IF SETNUM^= 4 THEN DELETE;
PROC RSQUARE DATA=NEW B;
MODEL Y = X3 X2              /INCLUDE=2;

FILENAME NEW '64.dat';
DATA NEW;
INFILE NEW;
INPUT SETNUM Y X1 X2 X3 X4 E1 E2 E3;
IF SETNUM^= 5 THEN DELETE;
PROC RSQUARE DATA=NEW B;
MODEL Y = X1 X2 X3 E1        /INCLUDE=4;
                    .
                    .
                    .
                    .
                    .
                    .

FILENAME NEW '64.dat';
DATA NEW;
INFILE NEW;
INPUT SETNUM Y X1 X2 X3 X4 E1 E2 E3;
```

```
IF SETNUM^=60 THEN DELETE;
PROC RSQUARE DATA=NEW B;
MODEL Y = X2 X1 X3            /INCLUDE=3;
```

```
;                          SAS Program PM.SAS
;
option linesize=80;
filename new 'PM.dat';
data new;
infile new;
input  DP ymse ysp ycp YMILLERS CONST A B AB C AC BC ABC D
AD BD ABD
       CD ACD BCD ABCD E AE BE ABE CE ACE BCE ABCE DE
       ADE BDE ABDE CDE ACDE BCDE ABCDE F AF BF ABF CF ACF
       BCF ABCF DF ADF BDF ABDF CDF ACDF BCDF ABCDF EF AEF
BEF
       ABEF CEF ACEF BCEF ABCEF DEF ADEF BDEF ABDEF CDEF
ACDEF BCDEF ABCDEF;


PROC PRINT;
TITLE 'Analysis of Performance Measures and Significant
Contributing Factors';
ID DP;
VAR ymse ysp ycp YMILLERS CONST A B AB C AC BC ABC D AD BD
ABD
       CD ACD BCD ABCD E AE BE ABE CE ACE BCE ABCE DE
       ADE BDE ABDE CDE ACDE BCDE ABCDE F AF BF ABF CF ACF
       BCF ABCF DF ADF BDF ABDF CDF ACDF BCDF ABCDF EF AEF
BEF
       ABEF CEF ACEF BCEF ABCEF DEF ADEF BDEF ABDEF CDEF
ACDEF BCDEF ABCDEF;


proc stepwise;
 model ymse = A B AB C AC BC ABC D AD BD ABD
       CD ACD BCD ABCD E AE BE ABE CE ACE BCE ABCE DE
       ADE BDE ABDE CDE ACDE BCDE ABCDE F AF BF ABF CF ACF
       BCF ABCF DF ADF BDF ABDF CDF ACDF BCDF ABCDF EF AEF
BEF
       ABEF CEF ACEF BCEF ABCEF DEF ADEF BDEF ABDEF CDEF
ACDEF BCDEF ABCDEF
       / stepwise slstay=.01;

proc stepwise;
 model ysp =  A B AB C AC BC ABC D AD BD ABD
       CD ACD BCD ABCD E AE BE ABE CE ACE BCE ABCE DE
       ADE BDE ABDE CDE ACDE BCDE ABCDE F AF BF ABF CF ACF
       BCF ABCF DF ADF BDF ABDF CDF ACDF BCDF ABCDF EF AEF
BEF
       ABEF CEF ACEF BCEF ABCEF DEF ADEF BDEF ABDEF CDEF
ACDEF BCDEF ABCDEF
       / stepwise slstay=.01;

proc stepwise;
 model ycp =  A B AB C AC BC ABC D AD BD ABD
```

```
       CD ACD BCD ABCD E AE BE ABE CE ACE BCE ABCE DE
       ADE BDE ABDE CDE ACDE BCDE ABCDE F AF BF ABF CF ACF
       BCF ABCF DF ADF BDF ABDF CDF ACDF BCDF ABCDF EF AEF
BEF
       ABEF CEF ACEF BCEF ABCEF DEF ADEF BDEF ABDEF CDEF
ACDEF BCDEF ABCDEF
       / stepwise slstay=.01;

proc stepwise;
 model YMILLERS = A B AB C AC BC ABC D AD BD ABD
       CD ACD BCD ABCD E AE BE ABE CE ACE BCE ABCE DE
       ADE BDE ABDE CDE ACDE BCDE ABCDE F AF BF ABF CF ACF
       BCF ABCF DF ADF BDF ABDF CDF ACDF BCDF ABCDF EF AEF
BEF
       ABEF CEF ACEF BCEF ABCEF DEF ADEF BDEF ABDEF CDEF
ACDEF BCDEF ABCDEF
       / stepwise slstay=.01;
```

```
;                    SAS Program STEP11_ALL.SAS
;
option linesize=80 pagesize=57;
filename new '01.dat';
data dataset;
infile new;
input  set y x1 x2 x3 x4 e1;
data randset;
 set dataset;
 r1=RANNOR(0);
 r2=RANNOR(0);
 r3=RANNOR(0);
 r4=RANNOR(0);
proc stepwise data=randset;
 by set;
 model y= x1 x2 x3 e1 r1 r2 r3 r4 /forward slentry=1;

filename new '03.dat';
data dataset;
infile new;
input  set y x1 x2 x3 x4 e1;
data randset;
 set dataset;
 r1=RANNOR(0);
 r2=RANNOR(0);
 r3=RANNOR(0);
 r4=RANNOR(0);
proc stepwise data=randset;
 by set;
 model y= x1 x2 x3 e1 r1 r2 r3 r4 /forward slentry=1;
.
.
.
.
.
.
.
filename new '31.dat';
data dataset;
infile new;
input  set y x1 x2 x3 x4 e1;
data randset;
 set dataset;
 r1=RANNOR(0);
 r2=RANNOR(0);
 r3=RANNOR(0);
 r4=RANNOR(0);
proc stepwise data=randset;
 by set;
 model y= x1 x2 x3 e1 r1 r2 r3 r4 /forward slentry=1;
```

```
;                    SAS Program STEP12_ALL.SAS
;
option linesize=80 pagesize=57;
filename new '33.dat';
data dataset;
infile new;
input  set y x1 x2 x3 x4 e1;
data randset;
 set dataset;
 r1=RANNOR(0);
 r2=RANNOR(0);
 r3=RANNOR(0);
 r4=RANNOR(0);
proc stepwise data=randset;
 by set;
 model y= x1 x2 x3 e1 r1 r2 r3 r4 /forward slentry=1;

filename new '35.dat';
data dataset;
infile new;
input  set y x1 x2 x3 x4 e1;
data randset;
 set dataset;
 r1=RANNOR(0);
 r2=RANNOR(0);
 r3=RANNOR(0);
 r4=RANNOR(0);
proc stepwise data=randset;
 by set;
 model y= x1 x2 x3 e1 r1 r2 r3 r4 /forward slentry=1;
.
.
.
.
.
.
.
filename new '63.dat';
data dataset;
infile new;
input  set y x1 x2 x3 x4 e1;
data randset;
 set dataset;
 r1=RANNOR(0);
 r2=RANNOR(0);
 r3=RANNOR(0);
 r4=RANNOR(0);
proc stepwise data=randset;
 by set;
 model y= x1 x2 x3 e1 r1 r2 r3 r4 /forward slentry=1;
```

```
;                SAS Program STEP31_ALL.SAS
;
option linesize=80 pagesize=57;
filename new '02.dat';
data dataset;
infile new;
input  set y x1 x2 x3 x4 e1 e2 e3;
data randset;
 set dataset;
 r1=RANNOR(0);
 r2=RANNOR(0);
 r3=RANNOR(0);
 r4=RANNOR(0);
 r5=RANNOR(0);
 r6=RANNOR(0);
proc stepwise data=randset;
 by set;
 model y= x1 x2 x3 e1 e2 e3 r1 r2 r3 r4 r5 r6 /forward
slentry=1;

filename new '04.dat';
data dataset;
infile new;
input  set y x1 x2 x3 x4 e1 e2 e3;
data randset;
 set dataset;
 r1=RANNOR(0);
 r2=RANNOR(0);
 r3=RANNOR(0);
 r4=RANNOR(0);
 r5=RANNOR(0);
 r6=RANNOR(0);
proc stepwise data=randset;
 by set;
 model y= x1 x2 x3 e1 e2 e3 r1 r2 r3 r4 r5 r6 /forward
slentry=1;
.
.
.
.
.
.
filename new '16.dat';
data dataset;
infile new;
input  set y x1 x2 x3 x4 e1 e2 e3;
data randset;
 set dataset;
 r1=RANNOR(0);
 r2=RANNOR(0);
 r3=RANNOR(0);
 r4=RANNOR(0);
```

```
 r5=RANNOR(0);
 r6=RANNOR(0);
proc stepwise data=randset;
 by set;
 model y= x1 x2 x3 e1 e2 e3 r1 r2 r3 r4 r5 r6 /forward
slentry=1;
```

```
;                    SAS Program STEP32_ALL.SAS
;
option linesize=80 pagesize=57;
filename new '18.dat';
data dataset;
infile new;
input  set y x1 x2 x3 x4 e1 e2 e3;
data randset;
 set dataset;
 r1=RANNOR(0);
 r2=RANNOR(0);
 r3=RANNOR(0);
 r4=RANNOR(0);
 r5=RANNOR(0);
 r6=RANNOR(0);
proc stepwise data=randset;
 by set;
 model y= x1 x2 x3 e1 e2 e3 r1 r2 r3 r4 r5 r6 /forward
slentry=1;

filename new '20.dat';
data dataset;
infile new;
input  set y x1 x2 x3 x4 e1 e2 e3;
data randset;
 set dataset;
 r1=RANNOR(0);
 r2=RANNOR(0);
 r3=RANNOR(0);
 r4=RANNOR(0);
 r5=RANNOR(0);
 r6=RANNOR(0);
proc stepwise data=randset;
 by set;
 model y= x1 x2 x3 e1 e2 e3 r1 r2 r3 r4 r5 r6 /forward
slentry=1;
.
.
.
.
.
.
.
filename new '32.dat';
data dataset;
infile new;
input  set y x1 x2 x3 x4 e1 e2 e3;
data randset;
 set dataset;
 r1=RANNOR(0);
 r2=RANNOR(0);
 r3=RANNOR(0);
```

```
 r4=RANNOR(0);
 r5=RANNOR(0);
 r6=RANNOR(0);
proc stepwise data=randset;
 by set;
 model y= x1 x2 x3 e1 e2 e3 r1 r2 r3 r4 r5 r6 /forward
slentry=1;
```

```
;                    SAS Program STEP33_ALL.SAS
;
option linesize=80 pagesize=57;
filename new '34.dat';
data dataset;
infile new;
input  set y x1 x2 x3 x4 e1 e2 e3;
data randset;
 set dataset;
 r1=RANNOR(0);
 r2=RANNOR(0);
 r3=RANNOR(0);
 r4=RANNOR(0);
 r5=RANNOR(0);
 r6=RANNOR(0);
proc stepwise data=randset;
 by set;
 model y= x1 x2 x3 e1 e2 e3 r1 r2 r3 r4 r5 r6 /forward
slentry=1;

filename new '36.dat';
data dataset;
infile new;
input  set y x1 x2 x3 x4 e1 e2 e3;
data randset;
 set dataset;
 r1=RANNOR(0);
 r2=RANNOR(0);
 r3=RANNOR(0);
 r4=RANNOR(0);
 r5=RANNOR(0);
 r6=RANNOR(0);
proc stepwise data=randset;
 by set;
 model y= x1 x2 x3 e1 e2 e3 r1 r2 r3 r4 r5 r6 /forward
slentry=1;
.
.
.
.
.
.
.
filename new '48.dat';
data dataset;
infile new;
input  set y x1 x2 x3 x4 e1 e2 e3;
data randset;
 set dataset;
 r1=RANNOR(0);
 r2=RANNOR(0);
 r3=RANNOR(0);
```

```
 r4=RANNOR(0);
 r5=RANNOR(0);
 r6=RANNOR(0);
proc stepwise data=randset;
 by set;
 model y= x1 x2 x3 e1 e2 e3 r1 r2 r3 r4 r5 r6 /forward
slentry=1;
```

```
;                    SAS Program STEP34_ALL.SAS
;
option linesize=80 pagesize=57;
filename new '50.dat';
data dataset;
infile new;
input  set y x1 x2 x3 x4 e1 e2 e3;
data randset;
 set dataset;
 r1=RANNOR(0);
 r2=RANNOR(0);
 r3=RANNOR(0);
 r4=RANNOR(0);
 r5=RANNOR(0);
 r6=RANNOR(0);
proc stepwise data=randset;
 by set;
 model y= x1 x2 x3 e1 e2 e3 r1 r2 r3 r4 r5 r6 /forward
slentry=1;

filename new '52.dat';
data dataset;
infile new;
input  set y x1 x2 x3 x4 e1 e2 e3;
data randset;
 set dataset;
 r1=RANNOR(0);
 r2=RANNOR(0);
 r3=RANNOR(0);
 r4=RANNOR(0);
 r5=RANNOR(0);
 r6=RANNOR(0);
proc stepwise data=randset;
 by set;
 model y= x1 x2 x3 e1 e2 e3 r1 r2 r3 r4 r5 r6 /forward
slentry=1;
.
.
.
.
.
.
.
.
filename new '64.dat';
data dataset;
infile new;
input  set y x1 x2 x3 x4 e1 e2 e3;
data randset;
 set dataset;
 r1=RANNOR(0);
 r2=RANNOR(0);
```

```
 r3=RANNOR(0);
 r4=RANNOR(0);
 r5=RANNOR(0);
 r6=RANNOR(0);
proc stepwise data=randset;
 by set;
 model y= x1 x2 x3 e1 e2 e3 r1 r2 r3 r4 r5 r6 /forward
slentry=1;
```

```
;                          SAS Program TM.SAS
;
option linesize=80;
filename new 'TM.dat';
data new;
infile new;
input  DP ymse ysp ycp YMILLERS CONST A B AB C AC BC ABC D
AD BD ABD
       CD ACD BCD ABCD E AE BE ABE CE ACE BCE ABCE DE
       ADE BDE ABDE CDE ACDE BCDE ABCDE F AF BF ABF CF ACF
       BCF ABCF DF ADF BDF ABDF CDF ACDF BCDF ABCDF EF AEF
BEF
       ABEF CEF ACEF BCEF ABCEF DEF ADEF BDEF ABDEF CDEF
ACDEF BCDEF ABCDEF;


PROC PRINT;
TITLE 'Analysis of Performance Measures and Significant
Contributing Factors';
ID DP;
VAR ymse ysp ycp YMILLERS CONST A B AB C AC BC ABC D AD BD
ABD
       CD ACD BCD ABCD E AE BE ABE CE ACE BCE ABCE DE
       ADE BDE ABDE CDE ACDE BCDE ABCDE F AF BF ABF CF ACF
       BCF ABCF DF ADF BDF ABDF CDF ACDF BCDF ABCDF EF AEF
BEF
       ABEF CEF ACEF BCEF ABCEF DEF ADEF BDEF ABDEF CDEF
ACDEF BCDEF ABCDEF;


proc stepwise;
 model ymse = A B AB C AC BC ABC D AD BD ABD
       CD ACD BCD ABCD E AE BE ABE CE ACE BCE ABCE DE
       ADE BDE ABDE CDE ACDE BCDE ABCDE F AF BF ABF CF ACF
       BCF ABCF DF ADF BDF ABDF CDF ACDF BCDF ABCDF EF AEF
BEF
       ABEF CEF ACEF BCEF ABCEF DEF ADEF BDEF ABDEF CDEF
ACDEF BCDEF ABCDEF
       / stepwise slstay=.01;

proc stepwise;
 model ysp =  A B AB C AC BC ABC D AD BD ABD
       CD ACD BCD ABCD E AE BE ABE CE ACE BCE ABCE DE
       ADE BDE ABDE CDE ACDE BCDE ABCDE F AF BF ABF CF ACF
       BCF ABCF DF ADF BDF ABDF CDF ACDF BCDF ABCDF EF AEF
BEF
       ABEF CEF ACEF BCEF ABCEF DEF ADEF BDEF ABDEF CDEF
ACDEF BCDEF ABCDEF
       / stepwise slstay=.01;

proc stepwise;
 model ycp =  A B AB C AC BC ABC D AD BD ABD
```

```
        CD ACD BCD ABCD E AE BE ABE CE ACE BCE ABCE DE
        ADE BDE ABDE CDE ACDE BCDE ABCDE F AF BF ABF CF ACF
        BCF ABCF DF ADF BDF ABDF CDF ACDF BCDF ABCDF EF AEF
BEF
        ABEF CEF ACEF BCEF ABCEF DEF ADEF BDEF ABDEF CDEF
ACDEF BCDEF ABCDEF
        / stepwise slstay=.01;

proc stepwise;
 model YMILLERS = A B AB C AC BC ABC D AD BD ABD
        CD ACD BCD ABCD E AE BE ABE CE ACE BCE ABCE DE
        ADE BDE ABDE CDE ACDE BCDE ABCDE F AF BF ABF CF ACF
        BCF ABCF DF ADF BDF ABDF CDF ACDF BCDF ABCDF EF AEF
BEF
        ABEF CEF ACEF BCEF ABCEF DEF ADEF BDEF ABDEF CDEF
ACDEF BCDEF ABCDEF
        / stepwise slstay=.01;
```

```
;                SAS Program TMSEP1_ALL.SAS
;
filename new '01.dat';
data new;
infile new;
input  set y x1 x2 x3 x4 e1  ;
proc rsquare data=new  mse sp cp b;
by set;
 model y= x1 x2 x3 e1  ;

filename new '03.dat';
data new;
infile new;
input  set y x1 x2 x3 x4 e1  ;
proc rsquare data=new  mse sp cp b;
by set;
 model y= x1 x2 x3 e1  ;

filename new '05.dat';
data new;
infile new;
input  set y x1 x2 x3 x4 e1  ;
proc rsquare data=new  mse sp cp b;
by set;
 model y= x1 x2 x3 e1  ;

filename new '07.dat';
data new;
infile new;
input  set y x1 x2 x3 x4 e1  ;
proc rsquare data=new  mse sp cp b;
by set;
 model y= x1 x2 x3 e1  ;

filename new '09.dat';
data new;
infile new;
input  set y x1 x2 x3 x4 e1  ;
proc rsquare data=new  mse sp cp b;
by set;
 model y= x1 x2 x3 e1  ;
.
.
.
.
.
.
.
filename new '63.dat';
data new;
infile new;
```

```
input  set y x1 x2 x3 x4 e1  ;
proc rsquare data=new  mse sp cp b;
by set;
 model y= x1 x2 x3 e1  ;
```

```
;                    SAS Program TMSEP3_ALL.SAS
;
filename new '02.dat';
data new;
infile new;
input  set y x1 x2 x3 x4 e1 e2 e3 @;
proc rsquare data=new  mse sp cp b;
by set;
 model y= x1 x2 x3 e1 e2 e3;

filename new '04.dat';
data new;
infile new;
input  set y x1 x2 x3 x4 e1 e2 e3 @;
proc rsquare data=new  mse sp cp b;
by set;
 model y= x1 x2 x3 e1 e2 e3;

filename new '06.dat';
data new;
infile new;
input  set y x1 x2 x3 x4 e1 e2 e3 @;
proc rsquare data=new  mse sp cp b;
by set;
 model y= x1 x2 x3 e1 e2 e3  ;

filename new '08.dat';
data new;
infile new;
input  set y x1 x2 x3 x4 e1 e2 e3 @;
proc rsquare data=new  mse sp cp b;
by set;
 model y= x1 x2 x3 e1 e2 e3  ;

filename new '10.dat';
data new;
infile new;
input  set y x1 x2 x3 x4 e1 e2 e3 @;
proc rsquare data=new  mse sp cp b;
by set;
 model y= x1 x2 x3 e1 e2 e3  ;
.
.
.
.
.
.
.
.
filename new '64.dat';
data new;
infile new;
```

```
input  set y x1 x2 x3 x4 e1 e2 e3 @;
proc rsquare data=new  mse sp cp b;
by set;
 model y= x1 x2 x3 e1 e2 e3  ;
```

Appendix K: Calculated Performance Measure Values for PM

DP	MSE_{pm}	$S_{p\ pm}$	$C_{p\ pm}$	$MILLERS_{pm}$
1	0.8483146	0.9230769	0.9121622	0.8721805
2	0.6313559	0.7272727	0.7127072	0.7578948
3	0.8560000	0.9203821	0.9102167	0.9187500
4	0.6569038	0.7414248	0.7315789	0.8616352
5	0.8682311	0.9304348	0.9178947	0.9230769
6	0.6528354	0.7266436	0.7217391	0.8141593
7	0.8748318	0.9348172	0.9255814	0.9259259
8	0.6569343	0.7335058	0.7278646	0.8471338
9	0.8721560	0.9282051	0.9188514	0.8688524
10	0.6557515	0.7311715	0.7254488	0.7983193
11	0.8755596	0.9261186	0.9178499	0.9322034
12	0.6597511	0.7431272	0.7348877	0.8418079
13	0.8723077	0.9229391	0.9143357	0.8769231
14	0.6631016	0.7472284	0.7362963	0.8174603
15	0.8725817	0.9200000	0.9112782	0.8994414
16	0.6633970	0.7509628	0.7395766	0.8545455
17	0.8766962	0.9201878	0.9125575	0.9352941
18	0.6724851	0.7601580	0.7501410	0.8983051
19	0.8786920	0.9198337	0.9120046	0.9179487
20	0.6803313	0.7726582	0.7640507	0.8800000
21	0.8789900	0.9225013	0.9134766	0.9076087
22	0.6854778	0.7806333	0.7723727	0.8518519
23	0.8792879	0.9256921	0.9169847	0.9322917
24	0.6923876	0.7895842	0.7820244	0.8620690
25	0.8801917	0.9274911	0.9190726	0.9470588
26	0.6963887	0.7929838	0.7863479	0.8444445
27	0.8800296	0.9282787	0.9197581	0.9270833
28	0.6977226	0.7966524	0.7905237	0.8421053
29	0.8809360	0.9288433	0.9205695	0.9180328
30	0.7002484	0.7979497	0.7921906	0.8410257
31	0.8810290	0.9304224	0.9220280	0.9090909
32	0.7022486	0.8001853	0.7948084	0.8900000
33	0.8808873	0.9289100	0.9209040	0.9345794
34	0.7007086	0.7967742	0.7916055	0.7619048
35	0.8794798	0.9267033	0.9188332	0.9057971
36	0.6968085	0.7930265	0.7888703	0.8120806
37	0.8795711	0.9254237	0.9174978	0.8983051
38	0.6940168	0.7899920	0.7851990	0.7377049
39	0.8784777	0.9255319	0.9174870	0.9071429
40	0.6909701	0.7874936	0.7823755	0.8214286
41	0.8779142	0.9241438	0.9163203	0.8938053
42	0.6885246	0.7851059	0.7803864	0.7614679
43	0.8772098	0.9234708	0.9151068	0.9127907
44	0.6867860	0.7835648	0.7789670	0.8654971
45	0.8757166	0.9216650	0.9139459	0.8770492
46	0.6866655	0.7838199	0.7796952	0.7777778

Calculated Performance Measure Values for PM (continued)

DP	MSE_{pm}	$S_{p\ pm}$	$C_{p\ pm}$	$MILLERS_{pm}$
47	0.8764809	0.9221646	0.9141683	0.9204546
48	0.6851399	0.7825346	0.7784675	0.8383234
49	0.8771561	0.9231497	0.9154701	0.9268293
50	0.6869755	0.7837176	0.7800328	0.8170732
51	0.8781572	0.9236174	0.9162409	0.9125683
52	0.6898949	0.7878609	0.7846002	0.9053254
53	0.8803269	0.9259421	0.9186071	0.9695122
54	0.6912658	0.7896428	0.7863248	0.8412699
55	0.8800600	0.9260113	0.9189467	0.9349113
56	0.6936138	0.7914847	0.7882611	0.8870968
57	0.8798051	0.9265981	0.9194219	0.9259259
58	0.6940785	0.7930974	0.7898628	0.8541667
59	0.8806620	0.9273645	0.9204350	0.9421053
60	0.6952754	0.7946640	0.7912852	0.9179487
61	0.8807000	0.9280630	0.9213628	0.9301075
62	0.6971050	0.7972907	0.7940019	0.8750000
63	0.8820463	0.9288945	0.9222420	0.9230769
64	0.6984938	0.7990220	0.7955985	0.8725491

Appendix L: Calculated Performance Measure Value for TMSEP

DP	MSE_{tm}	$S_{p\ tm}$	$C_{p\ tm}$	$MILLERS_{tm}$
1	0.7650620	0.8087910	1.0234350	1.1609520
2	0.6637320	0.7893480	1.0953370	1.6926820
3	0.1487610	0.1595710	0.2381690	0.1970290
4	0.1581330	0.1644490	0.2421660	0.2214360
5	0.8993290	0.9814150	1.2547690	1.3781180
6	0.6908280	0.7437650	1.2031360	1.4195740
7	0.1405690	0.1471160	0.2459100	0.1979690
8	0.1488380	0.1552430	0.2376890	0.2127430
9	77.9380340	83.1626890	108.2751310	132.1354680
10	56.1681590	61.7997280	98.8873060	135.7216490
11	10.2727160	10.7429670	20.8164410	12.9381190
12	7.9198800	9.1730830	20.6267190	15.4755950
13	75.7267150	82.3265460	113.1051330	131.7463990
14	53.3580280	61.4832120	107.3695450	141.4950870
15	9.5692500	9.8932590	19.2865920	12.2740480
16	8.0496980	8.8758070	18.0312250	16.5588720
17	0.9292540	0.9358480	1.4927760	1.1517850
18	0.8593490	0.8651500	1.4510400	1.0863170
19	0.1407100	0.1423830	0.2824660	0.1426570
20	0.1367260	0.1338730	0.2878020	0.1418870
21	0.9471070	0.9643650	1.4791970	1.0234790
22	0.8688670	0.8869560	1.5071520	1.0349690
23	0.1427590	0.1425130	0.2991620	0.1468790
24	0.1363060	0.1346770	0.2990880	0.1383570
25	89.9664610	90.8934100	147.2874150	101.9961320
26	84.1691510	85.6997380	141.4162900	102.0691220
27	12.1973890	12.3373200	26.9606130	12.7069930
28	10.9168790	11.2033390	26.4688630	12.9651400
29	91.0713040	91.3485720	142.1083830	101.7852550
30	84.5563960	85.3022690	144.7259980	98.6170500
31	12.4259120	12.5170760	28.0536590	12.4964480
32	11.2453370	11.4638000	27.5978010	12.0174870
33	0.9804800	1.1347960	1.3585030	1.5437970
34	0.8325220	0.8686400	1.1092190	1.4102540
35	0.2638300	0.2741530	0.2970570	0.3223040
36	0.3523780	0.3504670	0.3609280	0.3678640
37	0.9159700	0.9888550	1.3031770	1.4501740
38	0.9021820	0.9639130	1.1355340	1.3516600
39	0.2787260	0.2903530	0.3201970	0.3200920
40	0.3576060	0.3316960	0.3655560	0.3798990
41	68.9201740	74.7250060	108.4361880	124.8693010
42	57.7932780	65.6259770	97.4607010	132.9111790
43	9.4603400	9.7781960	18.6886480	13.1972780
44	8.4567630	9.0632910	20.8584960	15.7659790
45	79.2766190	83.3591160	107.4557270	119.1930390

Calculated Performance Measure Value for TMSEP

DP	MSE_{tm}	$S_{p\ tm}$	$C_{p\ tm}$	$MiLLERS_{tm}$
46	62.0207100	70.4859700	102.2720180	134.2141570
47	10.2950660	10.8312980	19.9675880	12.3031880
48	7.6850630	9.1260600	19.7178250	17.8853320
49	0.9873370	0.9887390	1.5157680	1.2343050
50	0.9494320	0.9554130	1.4946450	1.3705840
51	0.1949640	0.1986370	0.3434490	0.2177160
52	0.2108450	0.2073290	0.3380340	0.2504210
53	1.0238520	1.0291120	1.5982300	1.1669810
54	0.9318340	0.9543170	1.5202000	1.0758220
55	0.1930650	0.1973760	0.3198640	0.2236930
56	0.2103010	0.2086840	0.3424410	0.2333300
57	92.1139220	92.9549100	148.0592800	119.4541400
58	83.7433850	85.6070180	142.4074100	96.1285930
59	12.0630550	12.1251390	26.6133730	12.2637760
60	10.9248680	11.0991730	26.7097450	11.5297640
61	93.2993090	94.0328060	151.9314420	96.3185420
62	85.7757420	87.8176350	140.6162720	106.2438430
63	11.7405510	11.8084090	26.8109000	11.8390310
64	11.1539340	11.2962640	25.5540160	11.6543080

Bibliography

Barr, David R. "Interpretations of Mallows C_p Criterion in all Possible Regression Models." Unpublished report. AFIT/ENC, Wright-Patterson AFB OH, 1-7.

Berk, Kenneth N. "Comparing Subset Regression Procedures," Technometrics, 20:1-6 (February 1978).

Bremen,L. and D. Freedman. "How Many Variables Should Be Entered in a Regression Equation?" Journal of the American Statistical Association, 78: 1 31-6 (March 1983).

Cafarella, Joseph R., Jr. Cross Validation of Selection of Variables in Multiple Regression. MS Thesis, AFIT/GOR/MA/79D-2. School of Engineering, Air Force Institute of Technology (AU), Wright-Patterson AFB OH, December 1979.

Draper, Norman, and Harry Smith. Applied Regression Analysis, Second Edition New York: John Wiley & Sons, 1981.

Freedman, David A. "A Note on Screening Equations,"The American Statistician, 37:152-155 (May 1983).

Hansen, Capt Ross J. A Comparison of Variable Selection Criteria for Multiple Linear Regression: A Simulation Study. MS Thesis, AFIT/GOR/MA/88D-3. School of Engineering, Air Force Institute of Technology (AU), Wright-Patterson AFB OH, December 1988.

Healy, M.J. "The Use of R^2 as a Measure of Goodness of Fit" Journal of the Royal Statistical Society, Series A, 147: 608-609 (1984).

Hocking, R.R. "The Analysis and Selection of Variables in Linear Regression," Biometrika,32: 1-49 (March 1976).

-----, R. R. "Developments in Linear Regression Methodology: 1959-1982," Technometrics, 25: 219-230 (August 1983).

Judge, George, W.E. Griffiths, R. Carter Hill, Helmut Lutkepohl, and Tsoung-Chao Lee. The Theory and Prac-

tice of Econometrics, Second Edition. New York: John Wiley & Sons, Inc., 1985.

Miller, Alan J. "Selection of Subsets of Regression Variables," Journal of the Royal Statistical Society, Series A, 147,Part 3: 389-425 (1984).

-----, Alan J. Subset Selection in Regression. London: Chapman and Hall, 1990.

Narula, Subhash C. and John F. Wellington. "Selection of Variables in Linear Regression: A Pragmatic Approach," Journal of Statistical Computations and Simulation, 12:59-172 (1983).

Neter, John, William Wasserman, and Michael H. Ketner. Applied Linear Statistical Models. Homewood, IL: Richard D. Irwin, Inc., 1990.

SAS Institute Inc. SAS\ User's Guide: Statistics, Version 5 Edition. Cary, NC: SAS Institute Inc., 1985.

STATISTIX\ Version 4.0 User's Manual. Ed. Joan Siegel. St. Paul MN: Analytical Software, 1992.

Thompson, Mary L. "Selection of Variables in Multiple Regression: Part I. A Review and Evaluation," International Statistical Review. 46: 1-19 (1978).

Weisberg, Sanford. "A Statistic for Allocating C_p to Individual Cases," Technometrics, 23: 27-31 (February 1981).

Vita

Capt David P. Woollard was born 21 July 1961 in Dayton, Ohio. He graduated from high school in Miamisburg, Ohio, in 1979 and attended David Lipscomb University, Nashville, Tennessee. In 1983 he received a Bachelor of Science degree, majoring in mathematics and computer science. He joined the United States Air Force in 1986 and received his commission through Officer Training School. His first assignment was with the 2048th Communications Squadron, Carswell AFB. He held various leadership positions in the maintenance and operations branches with his last position being Chief of Communications Operations. In 1990, he completed his first Masters degree, obtaining a Master of Liberal Arts degree from Texas Christian University. He was reassigned to the Air Force Institute of Technology in May 1991.

He married the former Karen Ruth Troyer of Bowie, Maryland in 1983. They have two boys: Joshua, 7 years, and Jonathan, 21 months.

Permanent address: 3028 New Oak Lane
Bowie, Maryland 20716

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	March 1993	Master's Thesis

4. TITLE AND SUBTITLE
A COMPARISON OF VARIABLE SELECTION CRITERIA FOR MULTIPLE LINEAR REGRESSION: A SECOND SIMULATION STUDY

5. FUNDING NUMBERS

6. AUTHOR(S)
David P. Woollard, Capt, USAF

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Air Force Institute of Technology,
WPAFB, OH 45433-6583

8. PERFORMING ORGANIZATION REPORT NUMBER
AFIT/GOR/ENS/93M-23

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES
Approved for public release; distribution unlimited

12a. DISTRIBUTION/AVAILABILITY STATEMENT

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

This thesis implements a variable selection method proposed by Alan J. Miller, and makes an extension of Ross J. Hansen's 1988 thesis research by comparing the methods he examined: Minimum MSE, Minimum S_p, and Minimum C_p with Miller's method. Response Surface methodology is employed with two performance measures: the percentage of correct variables in a model and the Theoretical Mean Squared Error of Prediction (TMSEP). Each technique is applied on generated data with known multicollinearities, variances, random predictors, and sample sizes. Both performance measures are computed for models selected under each technique. A full factorial design using each performance measure is set up to study the effectiveness of each variable selection technique with respect to the known data characteristics. Equations are generated which relate these data characteristics to each combination of performance measure and selection method. A graphical analysis of variance is performed to summarize each technique's performance. Miller's method is shown to be the best overall technique for selecting models with the highest percentage of correct variables. Minimum MSE, followed closely by Minimum S_p, selected models with the least TMSEP.

14. SUBJECT TERMS	15. NUMBER OF PAGES
Statistics, Regression Analysis, Least Squares Method, Subset Selection	231
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	UL